Introduction

This is a summary of why our initial project failed, and we went with the supplied topics instead. The initial idea for this project was to use the Ashley Madison leaks by the Impact Team, and process the information to build a "cheater" profile. The dump contained a slew of other information, but it was low-density (the ALM office floor plan, employee reviews, banks and paypal accounts), highly uninteresting (a domain registry, server host hardware details), or outside the scope of an undergraduate project (password hashes for user and staff login credentials). If you remember a project from Henry Zhao in your class a few years ago, it might have been on the same data set (that I supplied).

User Profiles

The schema for user profiles included some juicy information, including full addresses, latitude and longitude, timestamps for both the accounts creation and their last access, relationship status, and what can I only assume to be reasonably accurate demographic information (lying about one's height, weight, age on a dating site tends to be difficult to pull off for long). It also included a less useful, disorganized set of "turn-on" flags, ranging from interests (i.e older/younger partners, discretion, open relationships) to fetishes and sex acts, which I was less inclined to read into.

The bulk of the analysis was going to be on these profiles, first by filtering out bots (see below) until we're confident in *most* of our data (which would then be, for the most part, a dataset of married, middle-aged men), and then using the profile usage data combined with billing information to build a profile of when people are most likely to cheat. This would be an aggregate of simple statistics, like when accounts are most likely to be made (lowest on Valentine's day, highest in fall/winter), and when payments/account activity is highest (early evenings, with some variance). Some of the things we'd need include:

- Discarding automated payments (such as the regular $51.45 fee)
- Identifying timezone based on city/coordinates
- Loose string matching to remove typos
- Flagging of "unused" accounts - those that are abandoned immediately after creation

I also wondered if I could spot a "schedule" based on the usage times of users with work numbers against those without, but the "work phone number" field was left untouched for nearly all active accounts.

Bot Filtering

The other interesting thing about the ALM data is that - as was later revealed - the site's female userbase was almost entirely populated by bots. A combination of matching email domains (especially those matching an existing ALM domain - almost certain a bot created by the company) and irregularities in account usage, like never checking one's messages while sending many, would have done a reasonable job filtering out bots. We never got to the rest.

What went wrong

To process the ALM leaks from SQL dump files, I used a python script to filter rows into csv format, and then a separate script to slice these into manageable files for spark. The naked python script wasn't very happy with processing >30GB of information (when expanded into readable form the whole set is something like 55 gigs), and when it came to slicing, had run out of memory and just... continued to write invalid chunks. I left these running overnight, and woke up to a bricked external hard drive (making that haunting knocking noise). The loss of the entire data set, most of my work, and the ability to even process that much data on a laptop means we're effectively starting over. Sorry Eric :(

Why am I telling you this?

This is mostly for my own sanity, and so I can look back and think "well at least he knows I wasn't a *boring* idiot". I feel better knowing my work was mediocre for lack of patience, work ethic, and care, rather than lack of ambition.

There's a lesson to be learned in here somewhere, and I hope that if nothing else, you pass it on to future students.