Summer Internship Review

Following is an outline/overview of the work I performed over the course of my summer internship 2018:

Usage Analytics

Mixpanel Wrangling

All user interactions are recorded in Mixpanel where detailed event logs provide information on content searched, features used, and problems encountered across all applications. Although Mixpanel provides several visualization and dashboarding tools, these tools are limited and somewhat difficult to use. The first step in this project was to liberate the data using Mixpanel's export API to bring the data into python scripts where proper analysis could be performed.

Results:

- **Python module** that will request a dataset from Mixpanel over a specified time range, and return the normalized dataset in a Pandas Dataframe.
- **FME Workspaces** to export Mixpanel data, filter/transform dataset, and upload to Postgres.

Next Steps:

- Clean up analytics repository (in Dashboards project) and move to new Business Intelligence project, organizing independent modules into separate repositories

Postgres Usage Schema

In order to make the Mixpanel data more accessible, I created a **new schema** (*usage*) in the Account-DB database to store usage information. This schema was created with **two tables**:

1. *product_event*: store all events exported from Mixpanel from FME jobs
2. *user_score*: store weekly calculated user health scores (discussed in more detail below)

Additionally, **two views** were created to add convenience:

1. *user_product_event*: implements SQL logic to add a valid user_id column to the product_event table based on the Mixpanel distinct_id (sometimes email, sometimes username, sometimes user_id)
2. *company_score*: aggregates average user health scores across each company to give overview of company usage

Lastly, the **`user` and `group` tables were altered** to add information about `job_type` and `group_type`. In the **`user`** table, `job_type_id` links to the **`job_type`** table which describes the roles that a client may hold in their company. This information will help better analyze usage. In the **`group`** table, `group_type_id` links to the **`group_type`** table which may describe the group as *Enterprise*, *SMB*, or *Oseberg* to facilitate analysis. These classifications were decided by Colin Murray and Brandon Sage based on account size (<$50,000 = SMB).

These additional classifications to user and group were added to the CSM Portal refresh project worked on by John Nguyen.

Next Steps:

- Implement changes in production account-db and run FME jobs to bring in historical mixpanel data
- Add to account-db for analytics Test
- Schedule FME jobs to regularly update the Postgres table with Mixpanel data

- Schedule FME Job for Mixpanel->PG ETL Open

User Health Model

After developing a robust set of usage statistics aggregated by user, I worked on building a model that would compress these statistics into a single "health score" representative of the user's engagement with the products.

More detail on the specific metrics used can be found on this confluence page or in this excel file. The following information is to highlight the assumptions/choices made during the development process

Of some 300+ statistics, 30 were chosen as relevant and important to usage. These focus on Atla usage primarily, but also include several core features of sol. Because each attribute is measured on a different scale, each value was normalized to a 0-1 range. To do this, continuous variables were reassigned by 10th quantile values (so that users in the top 10% of values will receive 0.9 points, and users in the bottom 30% will receive 0.3 points, etc.) Several variables were already brought in as percentages and thus left unchanged, boolean variables were converted to 1/0 for T/F. These 30 attributes were summarized and explained to the CSM/Sales teams who discussed and decided relative weights to assign. […] took lead on this and assigned each attribute a weight [1-5] based on importance. Those items ranked as a 5 were considered the most impactful or most difficult features to use so that users who performed these uncommon actions would be accordingly rewarded with higher score. Conversely, users that only interact with the common core features (ranked as 1's) will have lower scores.

Scores are currently calculated each week, examining a user's activity over the past week. This choice was made to highlight fluctuation and response to marketing campaigns, new features, CSM contact, etc.

In summary:

This model combines 30 aggregate usage statistics to produce a single super score for each user. The score is formatted on a percent scale and greatly favors users that use a wide variety of functionality across the products. Scores were also examined at a company level by averaging user scores for each group.

Caveats:

- To get a good feel for a user's health, it may be necessary to look at several weeks' scores so as to account for any special circumstances limiting usage for a particular week.
- A user may have a low score, even if they have made thousands of Atla queries in a given week, if querying Atla is the only action they have performed.
- Low scores that repeat week-to-week may be indicative of a user that has weekly reports or sol alerts being sent to them, but no activity otherwise.
- Company averages are significantly brought down by inactive users, and thus, large accounts with many licenses are shown to have very low scores.

Next Steps:

- Look at historical company scores and compare with Salesforce data to find a reasonable score threshold that may indicate a cancellation
- Complete implementation of Postgres migration and scheduling of ETL and score calculation jobs
- Schedule Python job for Analytics Health Scores Open
- Schedule Analytics Workflow Open

CHD County Doc Cleanup

New vendor brought in to supply lease documents from county courthouses. Index files uploaded to S3 in non-standard format. I worked to develop scripts for 5 counties (Andrews, Howard, Lea_NM, Reagan, and Crockett) as well as a template for all other counties to use.

Additionally, I wrote two python packages for use in this project. The first (*oseberg-py-s3utils*) is a library of tools to manage file interaction in S3. The second (*oseberg-maze-authority*) performed a suite of completeness checks against the standardized index records produced by each county script to guarantee data quality for the rest of the pipeline.

All of these county parsing scripts were set up to be run by operations in a consistent and simple manner. More information about the project and its usage can be found in the README in the project's repository. Usage information has also been documented on Confluence by the operations team.

This project involved a lot of cooperation among other team members, as well as extensive code review. Through this, I was able to develop a lot of skills around good version control and best practices in code readability and maintainability. This project also greatly increased my comfort and knowledge-base around programming in Python.

Next Steps:

- Ensure that all new scripts merged into develop are up-to-date with the `example.py` script which itself should be maintained with any updates or changes to general structure and style.
- Update CHD Scripts to match example template Open
- Work with QA to more thoroughly test parser scripts.
- Maintain *oseberg-py-s3utils* and *oseberg-maze-authority* packages in PyPi