Data Science Challenge for Ultimate Technologies Inc.

In this challenge, I explore logins, plan an experiment, and predict retained riders.

Part 1 - Explore login

The logins file contains about 93,000 logins collected over 3 1/2 months. To explore these logins, I aggregate them into 15-minute bins; visualize them by month, week, and hour; and decompose them into trend, seasonality, and residuals. In doing so, I observe the following:

Data quality problem: The logins purportedly start in 1970. The first month is January. The date starts with the 1st. Together, these point to date issues (maybe, an inappropriate assumption of Unix time?). I want to investigate dates as they are critical to this analysis. Going forward, I assume that the timestamps are correct in all but the year.

Mean, median, and distribution: Mean logins every 15 minutes is about 9.5. The median is about 7. The standard deviation is about 8. The distribution is not normal and has a positive skew. The distribution of the log of logins is not normal.

Overall trend: Overall, logins every 15 minutes are increasing from month to month, although logins drop in the last few weeks of this sample.

Weekly pattern: There is weekly "seasonality." During the week, the pattern of median logins is: flat on Monday and Tuesday at around 5; increases from Wednesday through Saturday up to aout 11, and a slight drop on Sunday back to 10.

Daily pattern: There is daily "seasonality." During the day, the hourly pattern of median logins is: a morning low from 6 am to 8 am; a midday spike; an afternoon decline; an evening pick up from 6 pm; a nighttime peak from 10 pm to 1 am; and a gradual decline into the early morning.

Decomposition: I decompose the data at different frequencies for different seasonalities using a simple additive model from statsmodels' "season_decompose" function. Graphing the weekly pattern from daily data over two months, I see the month's rising trend and the weekly seasonality from Monday to Saturday with a Sunday lull. Graphing the daily pattern from hourly data over ten days, I see the weekly growing trend and the daily seasonal spikes at midday and night.

Part 2 - Plan an experiment

To plan the described experiment, I do the following:

Articulate the goal: The goal is to increase the number of drivers available for weekday rides in Gotham and those available for weeknight rides in Metropolis. We want to encourage drivers from both cities to serve both complementary markets. If we meet this goal, we expect to improve our rider experience and business metrics. Of course, I validate my understanding of the goal with my stakeholder, the city manager.

Scope the experiment: I recommend we start small and focused on half of the problem, which is weekday rides in Gotham. Doing so, we might go quicker, apply our learnings sooner, incur costs in smaller increments, and reduce risk.

State the hypothesis and the null: The hypothesis is: if Ultimate reimburses drivers for weekday tolls between Gotham and Metropolis, more drivers are available for Gotham weekday rides. With more available drivers, riders wait less; riders are more satisfied; trips go up; and revenue goes up. The null hypothesis is: reimbursing for Gotham weekday tolls has no impact on driver availability.

Describe the primary metric: When trips are requested, the Ultimate app alerts nearby drivers, I assume. This is the basis for the primary metric of mean daily available drivers for Gotham weekday trips. I calculate this on a 5-day rolling basis to account for the difference in start-of-week and end-of-week traffic. This metric is aligned with this experiment's goal and hypothesis, is likely readily available, and is likely well understood at Ultimate.

Compare alternative primary metrics: Alternative metrics include: count of Gotham weekday pickups; count or percent of Gotham and Metropolis drivers with at least one Gotham weekday pickup; count or percent of Metropolis-focused drivers with at least 1 Gotham weekday pickup; and the number of weekday tolls reimbursed. However, these alternatives are not directly aligned with our goal, and some might be too complex to quickly understand.

Identify secondary metrics: To measure business impact, I proposed the following secondary metrics for Gotham weekday trips: mean rider wait time per trip, mean rider rating per trip, total trips, total revenue, and net review after tolls.

Validate metrics, data, and infrastructure: Before starting, I validate that our metrics are available (e.g., can I get counts of nearby drivers per trip?) and that the needed infrastructure is in place (e.g., can I assign drivers to test and control?). Ideally, I get my hands on existing pre-experiment data, do some exploratory analysis, and further improve my metric definitions.

Prepare the experiment: Next, I collaborate with the stakeholder to do the following: determine the size of the impact we want to measure (e.g., a 5% or a 20% improvement?); define the fraction of drivers I want bucket into control, test and not participating; do a power analysis to calculate the needed data; convert this into the days required to run the experiment; and randomly assign drivers. Plus, I collaborate with my stakeholder (and others, most likely) to make sure we have a clear reimbursement policy and a plan to communicate the policy to the relevant drivers.

Identify the test statistic: In this test, I compare the means of two samples from a population with an unknown standard deviation. I use a z-test statistic in a one-tailed test.

Interpret the results: After running the test for the defined period, I calculate the following: metrics for test and control, changes in test metrics as percents of control, and p-values. While I focus on the current values, I look back at changes over the course of the experiment. I conclude pass or fail. I do a similar analysis on secondary metrics as a group, adjusting the alpha to correct for multiple metrics. Plus, I explore why the metrics moved as they did.

Make a recommendation: I make a recommendation based on whether the experiment clearly succeeded, clearly failed, or, as often is the case, landed somewhere in between. I described the impact on secondary metrics and have a theory as to why we got these results. As we have just started these experiments, I might recommend expanding the scope of our experiments (e.g., focusing on Metropolis evening trips) and improving our metrics.

Part 3 - Predict retained riders

To predict retained riders, I do the following:

Calculate retained: I define a retained rider as one with a trip within the last 30 days. I assume the sample data was pulled on July 1, 2014, the latest date in the file. Accordingly, 18,804 riders--a bit less than 38% of the January cohort--are retained.

Clean, explore, and prepare: Before modeling, I convert date strings into datetimes; examine the data's shape, types, and nulls; and calculate some simple summaries. I do some feature engineering: replace categorical strings with numeric values; turn signup date into a signup weekday; fill numeric NaNs with median values; fill categorical NaNs with most-common values; eliminate "average surge" which is highly correlated with "surge percentage;" and add a column of random 1s and 0s as a point of comparison. Also, I drop "last trip date," which is similar to what I am predicting, and I want to validate my assumption that the other features are pulled from data that was known in only the first 30-days of the rider history. That is, I assume we want to predict 6-month retention after the first month of data.

Define my modeling strategy: I interpret this as a classification problem (e.g., map riders to retained or not retained). To do so, I start with a random forest classifier. Random forest models

frequently: work well across different data types; don't require scaling; are robust to outliers, non-linear data, and imbalanced sets; can be used with little training; and have low bias and moderate variance. My model isn't likely to be too slow as this is a tiny data set. However, random forest models might be hard to interpret.

Identify alternative approaches: There are many alternative classifiers. Common ones are: logistic regression, KNN, support vector machines, linear discriminant analysis, naive Bayes, and many more. At this point, I want to understand the problem and the data, establish a prediction baseline, and start a dialogue with the stakeholder. For example, it's not clear whether this is an exploratory problem (i.e., "help me understand what's important to rider retention") or a predictive problem (i.e., "accurately predict rider retention"). I'd take different approaches to these different problems.

Evaluate model performance: With little to no tuning, this model has a 76% accuracy on the test set and a similar out-of-bag error, which is the model's mean prediction error on training samples set aside in our bootstrapping. Right now, the model performs "just OK" with plenty of room for improvement if we further pursue this approach.

Examine feature importance: Random forest classifier's importance is the decrease in average Gini impurity across the forest of decision trees. Simply put, a feature with higher importance has more predictive power, contains more information and is more useful, all with respect to splitting riders into retained or not across the random forest.

Rank features by importance: This model indicates the most important feature is "average distance." Other high-importance features are "weekday percentage" and "driver's rating of ride." Medium-importance features are: "surge percentage,", "weekday of sign up," "city," and "rider's rating of driver." The least important features are: "phone OS" and "ultimate black users," which are only slightly more important than the random variable.

Start a conversation with stakeholders: Maybe, our most retainable riders are far-traveling, weekday, well-behaved riders? Random forest importance does not tell us the direction of any changes in features impact retention. Still, the random forest model let's share insights into our riders and learn what our stakeholder wants to understand.