Shipt – Interview Exercise

Introduction

There are four data sets provided with these questions. For the most part, the column names in each file are self-explanatory, but for "InterviewData_Activity.csv", here is a quick description of the variables:

- **userid** (our unique identifier of a member in the database)
- **date** the member last placed an order
- **age** (an integer value)
- **gender** ("M" or "F")
- **metropolitan_area** (sample metro areas)
- **device_type** used (Desktop, Tablet, or Mobile)
- **active** (the response variable and a binary outcome, "0" meaning they were not an active member and "1" meaning they were active at the time this illustrative data was generated)

Required Questions

Questions 1, 2, and 3 deal with "InterviewData_Cost.csv" and "InterviewData_Rev.csv".

(1) Using any functions/packages you want, join these two data sets by "date" and "source_id", returning all rows from both regardless of whether there is a match between the two data sets.

(2) Using any functions/packages you want, join these two data sets by "date" and "source_id", returning only the rows from the "Cost" file that have no corresponding date in the "Revenue" file.

(3) Using your result from #1, what are the Top 4 sources ("source_id" values) in terms of total revenue generation across this data set? How would you visualize the monthly revenue for those Top 4 sources? (note: you don't need to actually create a plot; you can just describe what your ideal visual would look like)

Questions 4 and 5 deal with "InterviewData_Activity.csv".

(4) Assuming you've read the data into a Pandas DataFrame called df, run the following code to build a basic logistic regression model:

```
>>> import pandas as pd
>>> import statsmodels.api as sm

>>> dummy_genders = pd.get_dummies(df['gender'], prefix = 'gender')
>>> dummy_metro = pd.get_dummies(df['metropolitan_area'], prefix = 'metro_area')
>>> dummy_device = pd.get_dummies(df['device_type'], prefix = 'device')

>>> cols_to_keep = ['active', 'age']
>>> activity_data = df[cols_to_keep].join(dummy_genders.ix[:, 'gender_M':])
>>> activity_data = activity_data.join(dummy_metro.ix[:, 'metro_area_Birmingham':])
```

```
>>> activity_data = activity_data.join(dummy_device.ix[:, 'device_Mobile':])
>>> activity_data = sm.add_constant(activity_data, prepend=False)
>>> explanatory_cols = activity_data.columns[1:]
>>> full_logit_model = sm.GLM(activity_data['active'],
                              activity_data[explanatory_cols],
                              family=sm.families.Binomial())
>>> result = full_logit_model.fit()
```

Apply this model to the same data that the model was trained on and assess the prediction accuracy.

(5) Split the data into training and test samples, and build a model over the training data using the following Python code:

```
>>> training_data = activity_data[1:4000]
>>> test_data = activity_data[4001:].copy()
>>> training_logit_model = sm.GLM(training_data['active'],
                                  training_data[explanatory_cols],
                                  family=sm.families.Binomial())
>>> training_result = training_logit_model.fit()
```

Assess the training data model's accuracy on the test data. Why does the accuracy change so much?

Question 6 deals with "InterviewData_Parsing.csv".

(6) This data comes from a subset of userdata JSON blobs stored in our database. Parse out the values (stored in the "data_to_parse" column) into four separate columns. So for example, the four additional columns for the first entry would have values of "N", "U", "A7", and "W". You can use any functions/packages you want for this.

Additional Questions – Pick One

Pick only one of these questions and answer it in less than 400 words.

A) Within our web and mobile apps, members can generally find items through search and/or the product category tree (note that you can also search after clicking into a product category, in which case the search is filtered by the chosen category). Let's say that we decide to test a different product category tree. The Product team asks for your help in setting up the test and calling the results. How would you help them: (i) figure out how long we should run this test; (ii) decide what metric to measure; (iii) and then evaluate the test?

B) One of the ways we attract new members is through digital marketing campaigns (e.g., on Facebook). Assume that we know a little bit about potential users who see an ad for Shipt on Facebook – things like name and general metropolitan area, and then can measure the impressions on the ad, clicks to our landing page, and then conversions on our landing page. Our goal then is to drive more conversions on the landing page. What are some ways you might look at the already collected data (or some ways to enrich the existing data set) to try and make recommendations to the Marketing team for how to

optimize their campaigns?

C) When we introduce new products or features, we generally prefer to implement them as a test at first – sometimes though, we don't have that option. Imagine that on the operations side, which deals primarily with shoppers, we decide to revamp the way in which potential orders (from a member) are offered to a shopper (they can either ignore the notification, decline it, or accept it). We've got 60 days of data under the old system and 60 days after the revamp was implemented, we're hoping to figure out whether it led to an increase in offer acceptance rate (proportion of accepts out of all the response options). How might you go about trying to quantify the change due to the revamp?