Berkeley School of Information

datascience@berkeley

Monetizing the Medallion

A Data-Driven Method of Increasing Cab Traffic

Anup Tirumala, Maxwell Lee, Sarah Cha, Tianhao Xu

Professor Edward Fine

Storing and Retrieving Data – w205

Presented on August 22, 2017

Identification of Business Opportunity

Public transportation has faced a wave of disruptive technologies in the past five years. Metro systems like subways and tollways have had to increase fares to support rising interest rate required on the revenue backed bonds, and conventional private transport like car rentals and services have lost market share to lower-cost, easier-accessed mobile apps like Uber. The New York City metropolitan area is a major hub for transportation services where almost all of the 11 million people are consumers. Cost of owning a car (exacerbated by parking expenses) as well as increasing traffic times and commuter costs have become a deterrent.

Subways and busses operated by the Metropolitan Transportation Authority (MTA) are still the predominant form of local transit, we believe there is an opportunity to aid the ailing taxi businesses. Taxis in NYC are regulated by New York Taxi and Limousine Commission, and a potential driver must purchase a medallion to legally drive within the city. Prices reportedly hit an all time high of over $1 million in 2014, but have steadily fallen over the last 3 years; reportedly one sold for $250k earlier this year.

The decrease in value of the medallion a direct symptom of evaporating market share. Consumers can find cheaper public alternatives which are often comparable in transit time, but the real detriment has been competitor fees. As an example, a ride from Manhattan to Newark airport runs almost $115 while the same trip on Uber can be between 33%-50% of the cost. While private-side competition is one area of possible improvement, we believe that there is more immediate benefit to winning some of the public transport market share. Earlier this year, New York governor Andrew Cuomo declared the NYC Subway system to be in a state of emergency giving mounting failures and delays.

As the NYC T&L Commission has revamped its technical platform, we believe that there is an opportunity for data to improve taxi usage Specifically, we would like to propose utilizing data from the MTA to display in real-time congestion so that taxi drivers can be more targeted in their driving. The premise is that given the poor subway conditions there is a possibility to siphon away new customers if a taxi can prove itself to be more available. If people are creatures of instant-gratification, then perhaps there are those that would be willing to forsake train delays and overcrowded cars for a more personal and enjoyable commute.

Data Description

Our primary data source is the MTA website and API. There are four kinds of data relevant for our purposes:

1. Subway Train Schedules described in GTFS Static format, refreshed periodically. (http://web.mta.info/developers/data/nyct/subway/google_transit.zip)

2. Station turnstile data, refreshed weekly. (http://web.mta.info/developers/turnstile.html)

3. Subway Trip Updates described in GTFS Realtime format, refreshed every 30 seconds. (http://datamine.mta.info/list-of-feeds)

4. Historical records of GTFS Realtime data stored every five minutes and going back several years. (http://web.mta.info/developers/turnstile.html)

The historical records of GTFS Realtime data is not very crucial for our business case, and the focus is more on the first three above. We consider the development of more predictive models in the further work section.

Data Architecture

The following is a list of the data architecture, indentation representing directory.

/w205 - master folder that project sits in

/README.md - document of instructions for work

/Turnstile - master folder for turnstile import

/README.txt - instructions for turnstile

/permissions.sh - script to permission user to run all files

/stations.sh - script to get static station/complex data

/get_historic_turnstile.sh - script to get historic data, default 5 previous

/hive_schema_on_read.sql - import base data into hive

/process_turnstile_data.sql - calculate the volume of entries at each subway stop during the previous 5 weeks.

/turnstileBatchUpdate - folder to add new week to historic data

/README.txt - instructions

/new_turnstile_week.sh - import new weekly file

/append_turnstile.sql - add to Hive table

/Twitter - functioning exercise2 that follows specific tweet; not currently used in any functionality

/Parse - master folder for parsing real-time feed

/README.txt - instructions for parsing

/.gitignore - denote ignore rules

/gtds-realtime.proto - GTFS protocol definitions

/nyt-subway.proto - NYCT subway extensions for basic GTFS

/parse_gtfsrt.py - parse GTFS-RT into csv

/hive_ddl.sql - create basic hive table structure

/Import_schedule.sh - get zip of schedule data

/schedule_data_on_read.sql - import schedule data into hive

/combine_stops_schedule.sql - combine stop and schedule table

/combine_real_time_data.sql - combine the real time data with the stop and schedule table, and the turnstile volume.

/gtfsrt_updates.sh - every 30 seconds receive new update

/Sarah - folder for visualization code

/r_shiny/subway_plot - final folder

/server.R - server file for app

/ui.R - user interface generation

Limitations of Chosen Architecture

While we believe the designed architecture and product are a first step in the direction of addressing the identified business problem, there are several areas where we know improvement could be made.

We coerce all data types to comma-separated and manipulate that raw data between local files, hdfs, and HIVE. While this helped us progress the project, we are aware that this is not resource efficient. This also limits us from incorporating a true, real-time structure. If we eventually tried to scale this to a higher volume and velocity source, it would induce latency. The constant writing back and forth also adds to this latency. Ideally, we would reduce processing time by making parsing more efficient using something like STORM.

The serving layer is also currently restricted to line updates with historical average coloring, restricting output to a posteriori facts. Specifically, we feel that in the null event of no delays, our output's value decreases. There is also some inconsistency in turnstile data quality and mapping to real time stations that

The PATH Forward

Two clear areas of improvement stand out to us. First, we believe that alternative data can be leveraged to improved ex ante outputs. As discussed above, we were able to manipulate the exercise 2 framework to track certain keywords or twitter accounts. Unfortunately, our specific filters often lead to empty spouts that did not add real value. However, we do believe that there is information to be gathered from social media, for example on pop events where crowds are gathering. Similarly, other sources like live weather data could augment our output's ability to predict not only congestion but consumer willingness to take private transport. Finally, as urban areas become denser, transportation infrastructure will likely begin to fail as it has in New York. There is a window of opportunity for a more-complete architecture to take inputs (GTFS, social media, weather, etc.) from other cities, and apply scale the application to identify possible revenue opportunities for taxi pickup.