Improving Taxi Revenue with Reinforcement Learning

Jingshu Wang[1] and Benjamin Lampert[2]

***Abstract*— In recent news there has been controversy surrounding Uber and Lyft taking business away from taxi drivers. In light of this controversy, we explored how machine learning could optimize a taxi drivers income by understanding the flow of demand throughout the day. We first used historical New York City (NYC) transaction data from 2013 to build summary statistics of the demand, fare, duration, and transition probabilities for different zipcodes and times across NYC. We then applied a reinforcement learning model to maximize the total revenue generated by an individual driver. Utilizing dynamic programming we backtested our model on 10k drivers and found that if they followed an optimal policy they would earn on average $\sim$20% more per day than following a non-optimal policy.**

I. INTRODUCTION

With the emergence of traditional taxi competitors such as Lyft and Uber, many New York Taxi cab drivers are concerned with unfair competition. These companies leverage analytics, mobile apps, and other data sources to adjust rates based on a wide set of features. Uber, for example, has a team dedicated to analyzing demand and usage across cities[1]. Providing similar services to normal drivers therefore would be an important step to allow them to compete. Our approach to this problem was first to analyze and preprocess a collection of 2013 taxi records. We used a linear model to understand which features best predict demand, and a reinforcement learning model to maximize an individual drivers income based on a set of summary statistics.

II. DATA

A. NYC Transactions

The dataset we used is from the 2013 NYC Taxi records made available by a Freedom of Information act and first published by Chris Whong[2]. The collection contains all 173 million taxi transactions for 2013, where each entry details the relevant information for each taxi trip. A summary of these features can be found in Table I.

Label	Example
pickup_datetime	2013/01/05 6:49:41 PM
dropoff_datetime	2013/01/05 6:54:23 PM
pickup_longitude	-74.004707
pickup_latitude	40.73777
dropoff_longitude	-74.009834
dropoff_latitude	40.726002
total_amount	$7

TABLE I

EXAMPLE OF SOME OF THE FEATURES CONTAINED IN THE RAW DATA.

[1]W. Jingshu jingshuw@stanford.edu
[2]B. Lampert lampertb@stanford.edu

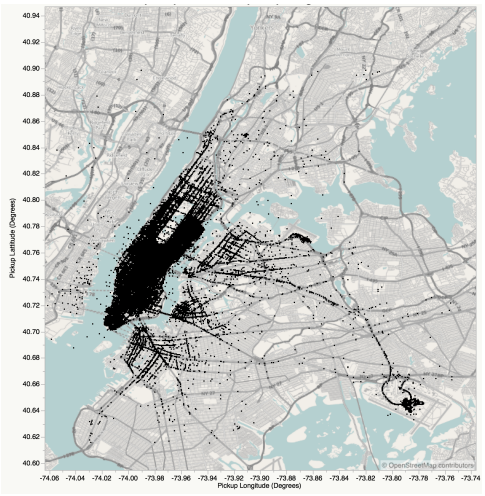

Fig. 1. Plot of each pickup location overlaid on a map of New York City. Each dot represents a transaction during a single day in January. Points of activity include Manhattan (left middle) and the JFK airport (lower right).

To handle the computational difficulties in training on large collections, we subsampled our data. We only examined transactions from January which resulted in a dataset of 14.5 million trips. For linear regression, we sampled and fit only 0.1% of the data. For reinforcement learning, we trained our model using the full dataset, but only backtested on sampled drivers from Wednesday records.

B. Preprocessing

To improve our location features we decided to bin the raw GPS coordinates of the pickup and drop-off location into a zipcode. This was accomplished using the python package Geodis[3]. We also applied a filter at this point to exclude all data not within NYC (ie, not reported correctly or did not successfully reverse geocode).

The next goal was to build a feature set that would allow us to train a policy on the data. We first iterated through all of the records to build a set of summary statistics related to the fare information. These data are grouped by the key (hour, pickupzipcode, dropoffzipcode). An example of one field from the summary data is as follows:

The next step was to get an estimate for the searching time for the next customer. Because our dataset only observes the time between pickup and drop-offs, we lack direct information related to how long the driver searched in an area to pick up a new customer.

To infer this data, we looked at drivers who picked up new customers in the same zipcode as where they had just dropped off a customer. This provided a time delta associated

Statistic	Value
Hour	10am
PickupZipCode	10012
DropOffZipCode	10011
Number of transistions	302
Mean Trip Distance	1.19 Miles
Mean Trip Duration	7.9 Minutes
Mean Fare	$8.58
Transition Probability	7.36%

TABLE II

EXAMPLE ENTRY IN THE SUMMARY STATISTIC DATA FOR PICKING UP A PASSENGER AT ZIPCODE 10012, WHO IS TRAVELING TO 10011, BETWEEN 10-11AM ON WEDNESDAY.

with how long they were searching between customers. We then modeled the searching time as an exponential distribution to get the expected waiting time for each zipcode and hour. More details will be found in Section III.

With these summary statistics we were then able to apply our model.

III. MODEL

A. Linear Regression

To find the relationship between taxi demand and various factors (geographical locations, time and date, weather etc.), we use a linear regression model to decompose the sources of variances. For the response, we aggregated the number of trips every three hours for each zipcode. For predictors, we consider zipcodes, the hours in a day, and the days in a week as three major factors along with their two-term interactions. We also include weather information during the 3 hour time interval as additional predictors. The model is:

$$\begin{aligned} Y_{ijk} &= \mu + \alpha_i + \beta_j + \gamma_k + (\alpha\beta)_{ij} + (\alpha\gamma)_{ik} + (\beta\gamma)_{jk} + \\ &\quad \sum_{s=1}^{p} a_s X_s + \epsilon_{ijk} \end{aligned}$$

where α_i are the main effects for zipcodes, β_j are the main effects for Monday, Tuesday, $\cdots$, Sunday, and γ_k are the 3 hour intervals for a day. $\{X_1, X_2, \cdots, X_p\}$ are the additional predictors of weather information.

B. Reinforcement Learning

1) Framework: When a taxi driver drops off a customer, and is looking for new business, there are two actions they can make. Either they can stay in the area and wait for passenger there, or travel to a new location. Travelling to a new location has an associated cost, but if chosen correctly the new location will impact future trips the revenue for the day. We find that a reinforcement learning model comes naturally to quantify the above optimization procedure. Using zipcodes as geographical units, we can define the states, reward and value functions as:

- States: (z, t) a dropoff zipcode at a corresponding time
- Reward: $R(z_1, t_1)$ average trip fair of one trip for a pickup zipcode z_1 at time t_1
- Value function: $V(z, t)$ expected total revenue to the end of the day starting from state (s, t)

In a traditional reinforcement learning model the intention of the driver would be an important feature. In practice, although a driver may intend to drive to a zipcode z, they can pick up customer along the way. However, it's impossible to know a driver's intention from historical data as we only know where he ends up picking the next customer. Thus, to reduce the difficulty in estimating the model parameters from historical data, we consider a simplified policy:

- Policy: $\pi(z, t) = z_1$ the next pickup zipcode from state (z, t). The driver goes to z_1 directly and will keep searching inside z_1 until he picks up the next customer at z_1

Then, we can write out the value function as:

$$\begin{aligned} & V^\pi(z, t) \\ = \; & E[f(\pi(z, t), Z', T_{\text{pick}}) + V^\pi(Z', T_{\text{next}})] \\ = \; & E\left[\sum_{z'}(P(z_1, z', T_{\text{pick}}) f(z_1, z', T_{\text{pick}}) + V^\pi(z', T_{\text{next}}))\right] \\ = \; & E\left[R(z_1, T_{\text{pick}}) + \sum_{z'} P(z_1, z', T_{\text{pick}}) V^\pi(z', T_{\text{next}})\right] \end{aligned}$$

where Z', T_{pick} and T_{next} represent the random zipcode that the customer at z_1 wants to go, the random pickup time at z_1 and the random dropoff time at Z' respectively. $f(z_1, z', T_{\text{pick}})$ is the trip fare from z_1 to z' at time t and $P(z_1, z', t)$ is a customers transition probability from z_1 to z'. The above equations hold after assuming that all the random variables are independent. To further quantify T_{pick} and T_{next}, we have

$$\begin{aligned} T_{\text{pick}} &= t + \Delta_{\text{travel}}(z, z_1) + \Delta_{\text{search}}(z_1) \\ T_{\text{next}} &= T_{\text{pick}} + \Delta_{\text{trip}}(z_1, z') \end{aligned}$$

where $\Delta_{\text{travel}}, \Delta_{\text{search}}$ and Δ_{trip} are random time intervals for the time cost traveling from z to z_1 without a passenger, searching for the next customer at z_1, and driving a passenger from z_1 to z' respectively. Finally, to approximate the expectation in (1), we replace all the random time with their expectations. As the value function is almost linear in t and our estimates of $R(z_1, t_1)$ and $P(z_1, z', t)$ will be piecewise constant (described later) in hour, this approximation should be accurate enough most of the time. We then define the optimal value function as:

$$V^\star(z, t) = \max_\pi V^\pi(z, t)$$

Then, it will satisfy:

$$\begin{aligned} & V^\star(z, t) \\ = \; & \max_{z_1}\left\{R(z_1, t_{\text{pick}}) + \sum_{z'} P(z_1, z', t_{\text{pick}}) V^\star(z', t_{\text{next}})\right\} \end{aligned}$$

where

$$\begin{aligned} t_{\text{pick}} &= t + \delta_{\text{travel}}(z, z_1, t) + \delta_{\text{search}}(z_1, t + \delta_{\text{travel}}(z, z_1, t)) \\ t_{\text{next}} &= t_{\text{pick}} + \delta_{\text{trip}}(z_1, z', t_{\text{pick}}) \end{aligned}$$

2) Estimating model parameters from historical data: To estimate the model parameters, we bin the data by days in a week and hours in a day. For $R(z,t)$, $P(z_1,z',t)$ and $\delta_{\text{trip}}(z_1,z',t)$, we simply take from historical data the average of the trip fare, the transit frequency and trip time inside the bin that t falls into. For δ_{travel}, we estimate it by:

$$\delta_{\text{travel}}(z,z_1,t) \quad = \quad \delta_{\text{trip}}(z,z_1,t) - \delta_{\text{trip}}(z_1,z_1,t)$$

as the driver will start searching as soon as he arrives at any point in z_1. Finally, we need to estimate δ_{search}. For this we need to consider that drivers may drive away to other zipcodes as they search for new passengers. Our prediction of $\delta_{\text{search}}(z_1,t_1)$ is defined as the expected searching time if the driver keeps searching in z_1 until he picks the next customer. Thus, simpily taking the average time duration of $z_{\text{dropoff}} = z^{\text{next}}_{\text{pickup}}$ to estimate δ_{search} will be a severally downward biased estimate.

Assume instead that Δ_{search} follows an exponential distribution $\exp(\lambda)$ which is a common distribution for waiting time. Then

$$\delta_{\text{search}}(z_1,t_1) \quad = \quad \frac{1}{\lambda(z_1,t_1)}$$

The exponential distribution has a nice property. If we can observe a lower quantile of Δ_{search}: $P(\Delta_{\text{search}} < x) = p$, then

$$\lambda = \frac{-\ln(1-p)}{x}$$

For the historical data, if the actual searching time was short, it's likely that the driver would still be in the same zipcode when he picks up the next customer, so we roughly observe a lower quantile of Δ_{search}, and we can determine a reasonable x as either the median or 75%(depending on how large they are) of duration times for $z_{\text{dropoff}} = z^{\text{next}}_{\text{pickup}}$ from data in the corresponding time bin. To estimate p, we assume that drivers are randomly searching, so if x is not too large, the drivers would still be in zipcodes close to z_1 and the searching time should be similar. Thus,

$$\hat{p}(z_1,t) = \frac{\#\{t^{\text{next}}_{\text{pickup}} - t_{\text{dropoff}} < x, z_{\text{dropoff}} = z_1\}}{\#\{z_{\text{dropoff}} = z_1\}}$$

3) Updating model parameters from future data: Updating our model parameters with online data can be accomplished by recalculating the summary statistics based on the new information. Obtaining feedback on a drivers intention would be an important new feature. Assuming that drivers follow our policy to maximize their revenue, we will know both where they intend to go and where they actually pick up the next customer. For a policy $\pi(z,t) = z_1$, instead of saying that the driver must pick up the next customer at z_1, we can estimate the transitions probabilities from z to a pickup zipcode z' for policy $\pi(z)$, and the new value function will be

$$\begin{aligned} & V^{\pi}(z,t) \\ = \quad & E\Bigg\{ \sum_{z''} P^{\pi=z_1}(z,z'',t) \Big[\sum_{z'} \\ & (P(z'',z',T_{\text{pick}}) f(z'',z',T_{\text{pick}}) + V^{\pi}(z',T_{\text{next}})) \Big] \Bigg\} \end{aligned}$$

Notice that P^{π} is the transition probability for the drivers while P is the transition probability for the customers.

IV. OUTCOME

A. Understanding the demand

The linear regression has $R^2 = 0.51$. The ANOVA table is:

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
PickupZip	94	9873.05	105.03	159.12	0.0000
day7	6	91.06	15.18	22.99	0.0000
hour	7	1188.95	169.85	257.32	0.0000
tempF	1	0.48	0.48	0.73	0.3931
windspeedMiles	1	0.01	0.01	0.01	0.9274
weatherDesc	10	28.75	2.87	4.35	0.0000
factor(winddir16point)	15	43.72	2.91	4.42	0.0000
precipMM	1	4.16	4.16	6.30	0.0121
humidity	1	0.03	0.03	0.04	0.8417
cloudcover	1	0.01	0.01	0.01	0.9265
HeatIndexF	1	1.71	1.71	2.59	0.1075
PickupZip:day7	564	757.07	1.34	2.03	0.0000
PickupZip:hour	658	2679.43	4.07	6.17	0.0000
day7:hour	42	293.88	7.00	10.60	0.0000
Residuals	22157	14625.13	0.66		

TABLE III

ANALYSIS OF VARIANCE: PICKUPZIP IS THE ZIPCODE FACTOR, DAY7 IS THE DAYS IN A WEEK FACTOR AND HOUR IS THE FACTOR FOR THE THREE-HOUR INTERVALS. OTHER FACTORS ARE INTERACTIONS AND WEATHER INFORMATION.

Some of the main effects and interaction are shown in Fig. 2.

B. Backtesting the optimal policy

The result of the reinforcement learning model is an optimal policy that details which zipcode mazimizes a drivers earning potential for the day. If a driver is looking for a new customer, the inputs of the policy would be his current zipcode and the time of day. The output would be the zipcode where he should look for a new passenger to maximize his revenue. Fig. 3 illustrates what such a policy looks like.

To back test our model we used transaction records to follow drivers for their shift. We compared their revenue for the day versus what they would have earned if they had used our policy. By using dynamic programming, we were able to back test over 10k drivers and record their performance with and without following our policy decision. Fig. 4 shows the performance of 200 representative drivers.

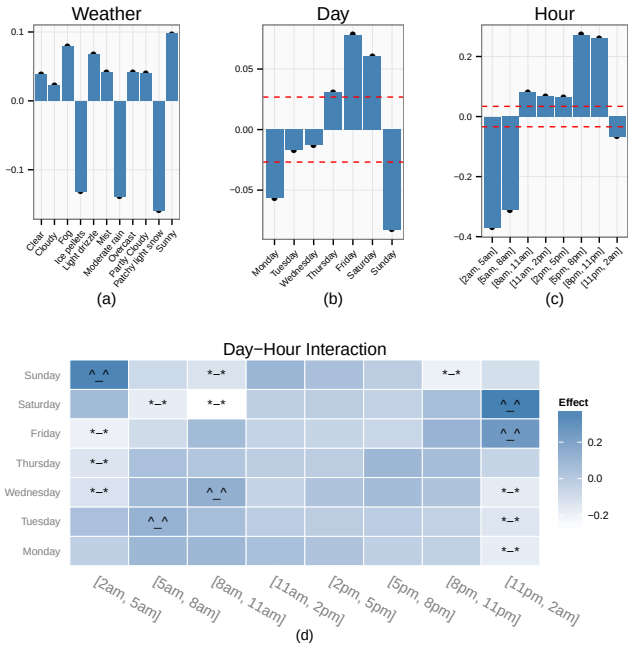

Fig. 2. Individual Effects:(a)-(c) are the main effects of variables weatherDesc, day7 and hour in Table III. For (b) and (c), the red dashed lines show the significance thresholds based on adjusted p-values. In (d) the color represents the estimate of each interaction effect. A smile face represents a significant positive effect and a sad face is a significant negative effect. All pvalues have been adjusted using Bonferroni to accommodate for multiple testing.

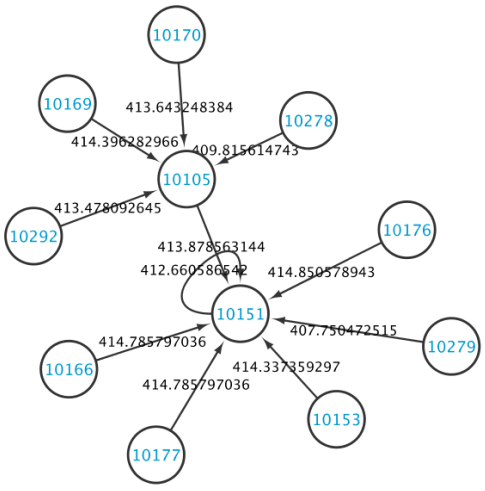

Fig. 3. Example policy from our model based on starting the day at 7am and ending the day at 5pm. Each node represents a starting zipcode, and the edges show the expected revenue for the day if the policy is taken.

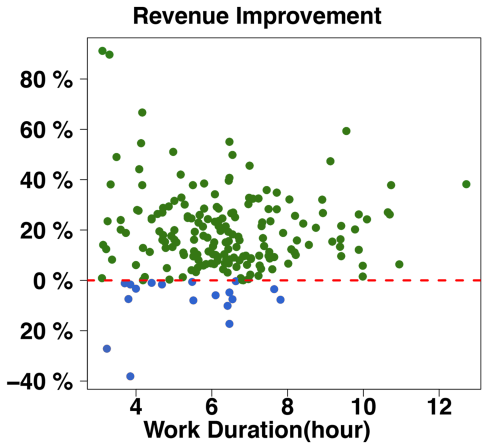

Fig. 4. Following the optimal policy from our model produced higher revenue than if a driver had used his own strategy. This result holds for a variety of work durations.

V. CONCLUSIONS

Applying a reinforcement learning model to the NYC taxi data we were able to show that an optimal policy can be obtained that maximizes the revenue generated by a single driver. Given a starting location and a time of the day our model can direct drivers to the optimal zipcode to find a customer. Backtesting results show that if a drivers were to follow our policy, they would earn ∼20% more revenue.

VI. FUTURE

There are finite number of passengers, and if all taxi drivers used this policy then they would not all see the same gains. Optimizing the entire system, and coordinating between all drivers would be a better approach and could ultimately benefit all parties. Taxi companies could allocate their medallions more efficiently, drivers would minimize their vacancy time and working hours, and passengers could see better Taxi coverage depending on peak hours/locations.

Another important piece of our model that is missing from traditional reinforcement learning algorithms is the feedback from making a decision. Tracking the intention of the driver to drive to a location, compared to where they end up, is an important piece of information. If this were to become a mobile app, then it might be possible to obtain this information and improve our model.

REFERENCES

[1] ”http://blog.uber.com/uberdata/” #uberdata Uber Blog. Uber, n.d. Web. 12 Dec. 2014.
[2] Whong, Chris. ”FOILing NYCs Taxi Trip Data.” FOILing NYCs Taxi Trip Data. N.p., 18 Mar. 2014. Web. 11 Dec. 2014.
[3] ”https://github.com/doat/geodis” Geodis. N.p., n.d. Web. 11 Dec. 2014.