Tracking Intensity of Ride-Sourcing Demand Using Mixture Models for Spatial Densities

Mauricio Garcia-Tec, Natalia Zuniga-Garcia

Contents

1. Introduction

Ride-sourcing companies, also known as transportation network companies (TNCs) or ride-hailing, provide pre-arrange or on-demand transportation service for compensation, which connect drivers of personal vehicles with passengers [1]. In recent years, TNCs such as Uber and Lyft, have grown exponentially. For instance, Lyft tripled its rides in 2016 [2] and Uber, which took six years to reach one billion trips, only took six months to reach the five billion milestone, on May 2017 [3]. However, there is a lack of understanding of the impact of these trips on the transportation network system. The main limitation is the availability of open-source data. Data-driven analysis can lead to evidence-based decision making and policies. Information such as the rides demand provides insights of economic activity, human behavior and mobility patterns in the cities.

TNCs have been involved in controversies in different cities around the world due to multiple factors, such as taxi service unfair competition, lack of regulation for their pricing system and driver selection, among others. The city of Austin is a clear example of these hassles. Uber and Lyft left the city in May 2016 after the Austin City Council passed an ordinance requiring ride-hailing companies to perform fingerprint background checks on drivers, a stipulation that already applies to Austin taxi companies [4]. Small companies, such as Ride Austin, Fasten, and Fare, started supplying the ride-sourcing demand. Ride Austin offered a new model of non-profit organization that also promotes the open-source data access.

The principal objective of this research is to provide a methodological framework to track the intensity of ride-sourcing demand using mixtures models. We used the data that Ride Austin made available for trips during the period that Uber and Lyft were out of the city, to evaluate the model. This information can provide valuable insights of the ride-sourcing trips in Austin.

The main contributions of this paper include: (1) a description of the principal mobility patterns of ride-sourcing trips in Austin, based on Ride Austin data; (2) a methodological framework to estimate demand intensity using mixtures models, including a R Package that users can utilize with similar datasets; and (3) an example of the application of the method to estimate hourly demand intensity maps for ride-sourcing trips.

The subsequent sections of the paper are organized as follow. Section 2 provides a general description of the dataset, representing main characteristics of the ride-sourcing trips in Austin. Section 3 presents the methodology used to develop a model for demand intensity using mixtures models. Section 4 explains the model implementation. Section 5 presents the results and an example of the application of the method to the ride-sourcing dataset available. Finally, the main findings are summarized in Section 6.

2. Ride-Sourcing Data Description

In this paper, we used the data that Ride Austin made available through the website `https://data.world/ride-austin`. The dataset consisted of 1,494,125 rides between June 2nd, 2016 and April 13th, 2017. Each trip corresponds to a row in the database. In addition, the dataset provides a description of the trip, rider and driver (anonymized), payment, cost, and weather. Table A.1, presented in the Annex section, provides a list of all the variables included in the dataset. We selected rides with the origin and destination coordinates within the traffic analysis zones (TAZ) as defined by the Capital Area Metropolitan Planning Organization (CAMPO). Figure 1 presents the area of study.

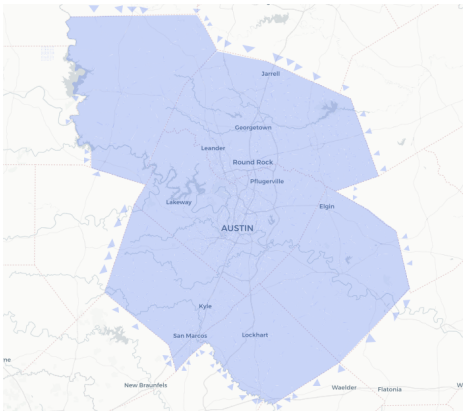


Figure 1: Area of Study

The data contains the main ride-sourcing trips patterns in the city given that the principal TNCs companies were not operating during the specified time frame. Therefore, its contributions can explain the general behavior of common ride-sourcing users in Austin. Figure 2 presents a description of the trips patterns found. We can observe that the company's trips presented an exponential growth from June to October 2016, as shown in Figure 2 (a). After this initial growth, it remains approximately constant. The average number of trips per day during the period of analysis is 4,757. The trips present a high hourly and daily variation. The hourly trips, shown in Figure 2 (b), present a variation range from 1 to 1,548 trips, with an average of 205 trips per hour.

Figure 2 (c) shows the number of trips variation during one week in April 2017. This graph allows observing in detail the hourly and daily variation of trips request. We can observe that the majority of the trips are concentrated during the night and early morning, corresponding to recreational trips. In addition, Monday (April 3rd) to Thursday (April 6th) show a similar number of trips, while Friday (April 7th) presents a significant increment. Saturday (morning and night) and Sunday (morning) present the larger peaks of the week. During Sunday night the trips decrease again. Figures 2 (d) to (f) provide a summary

of the total trips per weekday and per hour during weekday and weekend. We can observe that the behavior described previously for the selected week in April can be generalized as a common behavior of trips. It is important to mention that the peak hours of the ride-sourcing trips differ significantly with the typical peak hours of the system, which implies that this kind of trips do not increase congestion, and operate mainly in hours where the capacity is not restricted, such as early morning hours.

Additionally, the database showed that ride-sourcing trips tend to be short and inexpensive trips. The distribution of trip distance is presented in Figure 2 (g), where the mean and median distances are 8.4 and 5.8 km, respectively. The total trip fare distribution is presented in Figure 2 (h), the mean and median total costs are $13.7 and $11.1, respectively.

3. Model Development

We are interested in dynamically tracking the density and intensity of trips originating within the Austin area, as well as the distribution of quantities that vary smoothly across the region, such as the cost, distance, and duration of a trip. We approach this problem using the framework of spatial Poisson point processes, which we describe in the next section. The especially nice feature of this methodology is that enables us to separate the problem into the three independent subproblems of estimating: (1) a spatial density; (2) a global intensity function, (3) space-dependent regressions.

3.1. Counts of events as Poisson processes

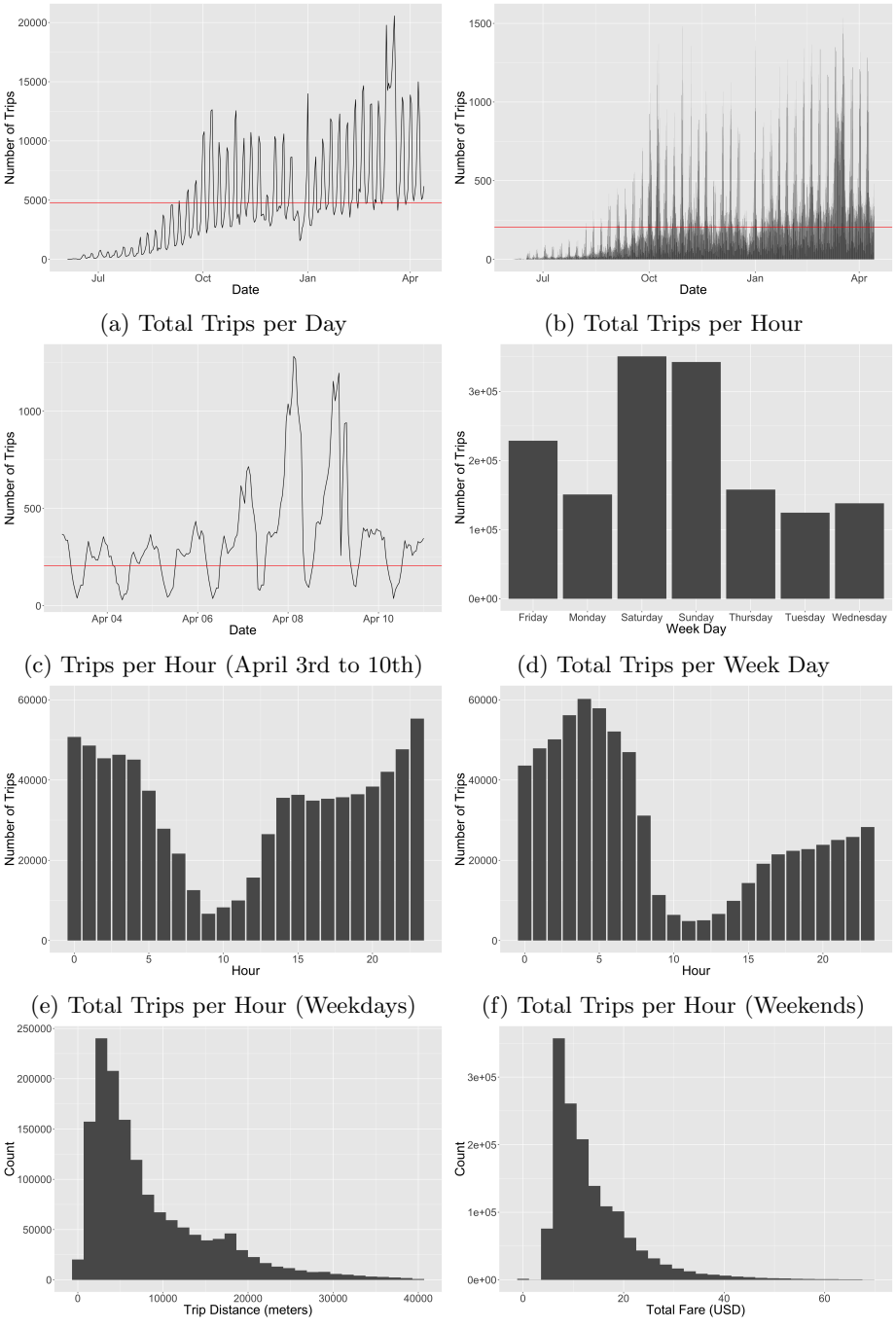
In this section we develop a *static time* model for ride-sharing trips. Suppose that for fixed space-window S and time-window T we observed n trips, which we denote as $\{x_i\}_{i=1}^n$. Each x_i here is a vector carrying the information related to the trip; typically, $x_i \in \mathbb{R}^2$ will be the longitude and latitude of the origin of the trip, although the destination could also be included. Other non-spatial quantities of interest, such as the cost, and traveling distance and time, will be modeled separately as we shall justify later.

The *Poisson point-process assumption* states that for any subregion of space $R \subset S$, the number of observed points in that region $\#(R)$ is distributed as a Poisson random variable

$$\#(R) \sim \mathrm{Po}(\Lambda(R)), \qquad \Lambda(R) = \int_R \lambda(s)\,\mathrm{d}s$$

where $\lambda\colon R \to [0, \infty)$ is a *local intensity function* we have to estimate. The total intensity over the whole space-window S is denoted $\Lambda = \Lambda(S)$ for simplicity.

One critical assumption of the Poisson approach is that the occurrences of points of non-overlapping regions are independent, that is, $Q \cap R = \emptyset$ implies $\#(Q)$ is independent of $\#(R)$ (conditional on knowing the intensity function). This is rarely a highly restrictive assumption since it does not mean that $\#(Q)$ and $\#(R)$ shouldn't have similar values, but that the randomness observed



(a) Total Trips per Day

(b) Total Trips per Hour

(c) Trips per Hour (April 3rd to 10th)

(d) Total Trips per Week Day

(e) Total Trips per Hour (Weekdays)

(f) Total Trips per Hour (Weekends)

(g) Trip Distance

(h) Total Trip Fare

Figure 2: Trips Description

does not depend on some unobserved quantity, which is safe to assume in our problem.

The name *point-process* comes from the fact that there is a spatial indexing variable, in analogy with the traditional time-indexed stochastic processes in probability theory.

The following fact about Poisson processes will be fundamental: conditional of having observed n points in S, the location of the observations are distributed independently proportional to λ, that is

$$\mathrm{p}\big(\{x_i\}_{i=1}^n \mid \#(S) = n, \Lambda, f\big) = \prod_{i=1}^{n} f(x_i), \qquad f(x) = \lambda(x)/\Lambda.$$

Moreover, instead of trying to directly estimate the local intensity function λ, we can think of the density function f and Λ as the unknown parameters, which leads to a nice decomposition of the likelihood function for our observed process [5].

$$\begin{aligned} L(\Lambda, f) &= \mathrm{p}\big(\{x_i\}_{i=1}^n \mid \Lambda, f\big) \\ &= \mathrm{p}\big(\#(S) = n \mid \Lambda, f\big)\, \mathrm{p}\big(\{x_i\}_{i=1}^n \mid \#(S) = n, \Lambda, f\big) \\ &= \mathrm{Po}\,(n; \Lambda) \textstyle\prod_{i=1}^n f(x_i) \\ &\propto \Lambda^n \exp\{-\Lambda\} \textstyle\prod_{i=1}^n f(x_i). \end{aligned}$$

Another consequence of this formulation is that the statistical inference procedures for estimating the *intensity scaling factor* Λ and the normalized density function f can be done independently, using appropriate statistical techniques for each case.

Suppose now that the observations $\{x_i\}_{i=1}^n$ are not completely identical, but instead they are "marked", each one being accompanied by a corresponding information or marking vector $\{y_i\}_{i=1}^n$. These markings can be discrete (e.g., type of service) or continuous (e.g., cost, traveling time, total distance). We can still treat all the occurrences of x as homogeneous and independently make a predictive model for y given x. This is true since

$$\mathrm{p}(x, y) = \mathrm{p}\big(y \mid x\big)\, \mathrm{p}(x)\,.$$

Hence we will focus on modeling the $\mathrm{p}(x)$ as a point-process, and then estimate $\mathrm{p}\big(y \mid x\big)$ with more appropriate separate techniques. In the rest of this section, we focus on the problem of estimating the density $\mathrm{p}(x)$, using non-parametric Bayesian techniques.

3.2. Motivation from finite mixture models

A popular approach for estimating densities is to use mixtures of normal distributions

$$x \sim \sum_{j=1}^{m} \omega_j N(\mu_j, \Sigma_j)$$

for some unknown weights $\omega_j \geq 0$ with $\sum_{i=1}^m \omega_j = 1$. The general assumption is to assume that for each observation x_i, there is an unobserved allocation variable k_i such that

$$\mathrm{p}\big(x_i \mid k_i = j\big) = N(x_i; \mu_j, \Sigma_j).$$

The unknown distribution of the k_i is of course related to the weights $\{\omega_j\}_{j=1}^m$. This approach is very powerful and can approximate many density functions.

In order to do Bayesian modeling, it is necessary to assume a prior distribution over the unknown parameters. This assumption, of course, has the advantage of allowing to incorporate previous information. A standard choice of prior for the unknown clusters (μ_j, Σ_j) is to suppose they are independent and identically distributed according to the Normal-Inverse Wishart prior

$$\mu_j \mid \Sigma_j \overset{iid}{\sim} N(\lambda_0, \Sigma_j/\kappa_0) \quad \text{and} \quad \Sigma_j \sim IW(\Omega_0, \nu_0),$$

with hyperparameters $\kappa_0, \nu_0, \Omega_0$ (the underscore zero is used to emphasize they are fixed). Under this specification we have

$$\mathbb{E}[\mu] = \lambda_0 \quad \text{and} \quad \mathbb{E}[\Sigma^{-1}] = \nu_0 \Omega_0^{-1}.$$

The less known Inverse Wishart distribution generalizes the inverse Gamma distribution and always yields a proper covariance matrix (a positive definite matrix); it is a popular choice in Bayesian modeling since it leads to a conjugate posterior, which is the Bayesian jargon for saying that the posterior distribution of (μ_j, Σ_j) after observing $\{x_i\}_{i=1}^n$ will still be a Normal-Inverse Wishart pair with updated (data-dependent) parameters. This greatly reduces the computing complexity, while still giving sufficient liberty for specifying prior belief on the location and precision of prior knowledge.

More complicated models can be specified with differing prior means and its dispersion for each cluster, but we prefer this parsimonious approach, which is also more easily generalized to infinite dimensions, as we do in the next section.

We still need to specify the prior of the weights ω_i and the unobserved allocation variables k_i. Our choice here will be the Dirichlet-Multinomial pair

$$\{\omega\}_{j=1}^m \sim Dir(\alpha_0/m) \quad \text{and} \quad k_i \overset{iid}{\sim} Mult\left(\{\omega\}_{j=1}^m\right).$$

Here, the Dirichlet distribution is a probability distribution over discrete probabilities; the output is a list of m non-negative numbers that add up to one. The hyperparameter $\alpha_0 > 0$ indicates a concentration of probabilities, the closer it is to zero, the more similar the output probabilities will be, and as it goes to infinity, it will output probabilities that have one entry close to one and the rest close to zero. The multinomial distribution simply picks an integer j with probability ω_j.

For strong convenience and theoretical reasons, it will be useful to rewrite the above formulation slightly different. We will think of the Dirichlet prior on the weights and the Normal mixture components (μ_j, Σ_j) together. More precisely,

we will say that (μ, Σ) is a single random variable distributed according to the following discrete probability function[1]

$$p(\mu, \Sigma) = \sum_{i=1}^{m} \omega_j \delta_{(\mu_j, \Sigma_j)}(\mu, \Sigma),$$

which is simply saying that (μ, Σ) takes the value (μ_j, Σ_j) with probability ω_j. The big win of doing this is that now we can write the model as

$$p(x) = \int N(x; \mu, \Sigma) p(\mathrm{d}(\mu, \Sigma)),$$

which will lead to the standard formulation of the non-parametric form. The ω_j here are the weights obtained from the Dirichlet distribution; the (μ_j, Σ_j) are still regarded as draws from the prior distribution

$$G_0(\mu, \Sigma) = N(\mu; \lambda_0, \Sigma/\kappa_0) IW(\Sigma; \Omega_0, \nu_0)$$

convenient for conjugate Bayesian analysis.

3.3. Non-parametric mixture models

An inconvenience of finite mixture model of the preceding section is the need to try different values of m, the number of clusters. We will instead take a non-parametric approach and suppose that the density function has the form

$$x \sim \sum_{j=1}^{\infty} \omega_j N(\mu_j, \Sigma_j),$$

The advantage of doing this is to obtain a more flexible with a number of parameters increasing as more data becomes available. Intuitively, the idea will be to add the data sequentially and start with one cluster, letting the data decide when should a new cluster be added if the current estimation of clusters does not fit a new batch of data properly.

Infinity does not really exist in any machine, and we need to find clever ways represent it. Here, we will need to assume that up to certain time only a finite number of cluster had been assigned, but there is still remaining probability to assume to new ones if necessary.

The Dirichlet Processes is a standard tool in Bayesian non-parametric modeling that extends the Dirichlet distribution to infinite dimension, always with respect to an underlying probability function G_0. The need of G_0 is part of the technicalities in its construction.

The construction of the Dirichlet Process can be extremely technical and cumbersome, so we will not develop it here, and leave it to the reader. A useful

[1] δ denotes the Dirac measure.

representation of the Dirichlet Process is the so-called *stick-breaking weights* representation.

We will say that (μ, Σ) are distributed $DP(\alpha_0, G_0)$, meaning that

$$p(\mu, \Sigma) = \sum_{j=1}^{\infty} \omega_j \delta_{(\mu_j, \Sigma_j)}(\mu, \Sigma),$$

with weights ω_j obtained from the *stick-breaking* generating model

$$\omega_j = \eta_j \prod_{l<j}(1-\omega_l), \quad \eta_j \overset{iid}{\sim} Beta(1, \alpha_0) \quad j = 1, \ldots, \infty.$$

and atoms (μ_j, Σ_j) from the prior distribution

$$G_0(\mu, \Sigma) = N(\mu; \lambda_0, \Sigma/\kappa_0) IW(\Sigma; \Omega_0, \nu_0).$$

Finally, our infinite mixture model for the spatial data x writes as a generalization of the finite case

$$p(x) = \int N(x; \mu, \Sigma) p(\mathrm{d}(\mu, \Sigma)) = \sum_{j=1}^{\infty} \omega_j N(x; \mu_j, \Omega_j).$$

The idea of stick-breaking is starting with a stick of length one, and at each time j, a new weight ω_j is obtained proportionally to η_j times the length of the current remaining stick, leaving the rest of the stick for future weights. This will create a sequence that in the limit will add up to one. We note that one feature of the stick-breaking process is that the orders have a form of probabilistic order (they are not perfectly ordered, but decaying).

3.4. Dynamic mixture model

Since we are interested in tracking a density over time, we use a dynamic model on the weights of the Dirichlet Process. As a consequence, our mixture will have the form

$$p(x) = \sum_{j=1}^{\infty} \omega_{j,t} N(x; \mu_j, \Omega_j).$$

We note that only the weights follow a dynamic process. The strategy we use is based on [5]. We assume the following autoregressive form for stick-breaking weights ω_j, depending on a fixed autoregressive dependence parameter $0 < \rho_0 < 1$

$$\omega_{j,t} = \eta_{j,t} \prod_{l<j}(1-\omega_{l,t})$$

where

$$\eta_{j,t} = 1 - u_t(1 - v_t \eta_{j,t-1}) \quad u_t \sim Beta(\alpha_0, 1-\rho_0),\ v_t \sim Beta(\rho_0, 1-\rho_0).$$

We emphasize that under this definition the $\eta_{j,t} \sim Beta(1, \alpha_0)$ for all t, so we still have a Dirichlet Process for each time t. The design is chosen so that the autocorrelation of the process satisfies

$$\operatorname{corr}(\eta_{j,t}, \eta_{j,t-1}) = \frac{\alpha_0 \rho_0}{1 + \alpha_0 - \rho_0}.$$

In practice, this autoregressive density estimation works considerably better that running independent density estimations, since it filters outs some of the noise. It also seems to help diminish the dependency on the hyperparameters.

To concisely write this autoregressive model, we put it in terms of a prior and say that

$$\text{at a given time period } t\text{:} \quad (\mu, \Sigma) \sim G_t, \quad G_t \sim DP(G_0, \alpha_0),$$

$$\text{for all periods } t = 1, \ldots, T\text{:} \quad \{G_t\}_{t=1}^{T} \sim BAR(\alpha_0, \rho_0).$$

The notation BAR stands for Beta Autoregressive Stick-Breaking model [6].

3.5. Particle filters

We chose particle filters and the main method because it is an *online* inference method [7]. We only superficially discuss the overall purpose of the method, details are left to the reader to be consulted in [7] or [5]. The following is a general form of the algorithm based for learning parameters θ within a model based on data x (in our mixture problem, $\theta = \{(\omega_j, \mu_j, \Sigma_j)\}_{j=1}^{\infty}$).

- **Initialize**. Start with a population of N_0 particles $\{z_0^i\}_{i=1}^{N_0}$. Each particle z_0^i is a list of information carrying *sufficient statistics*[2] (later on we specify the sufficient statistics we'll need). The particles state at time z_t must be sufficient for:
  - predicting: $\mathrm{p}\big(x_{t+1} \mid x_1, \ldots, x_t\big) = \mathrm{p}\big(x_{t+1} \mid z_t\big)$.
  - updating: $\mathrm{p}\big(z_{t+1} \mid x_1, \ldots, x_{t+1}\big) = \mathrm{p}\big(z_{t+1} \mid x_{t+1}, z_t\big)$.
  - learning: $\mathrm{p}\big(\theta \mid x_1, \ldots, x_{t+1}\big) = \mathrm{p}\big(\theta \mid z_{t+1}\big)$.
- For each new data point x_t:
  1. **Smoothing (resampling)**. Let the fittest survive! Resample each particle according the probability of the new data point given the current state of each particle. The resampling is achieved from

$$\gamma_k = \frac{\mathrm{p}\big(x_{t+1} \mid z_t^k\big)}{\sum_{l=1}^{N_0} \mathrm{p}\big(x_{t+1} \mid z_t^l\big)}, \quad k = 1, \ldots, N_0$$

[2] For example, sufficient statistics for fitting a simple normal distribution are the mean and sum of squares.

$$\zeta_i \overset{iid}{\sim} Cat(\gamma, N_0) \quad z_t^i \leftarrow z_t^{\zeta_i}, \quad i = 1, \ldots, N_0$$

A motivation for this step is Bayes' rule

$$\mathrm{p}\big(z_t \mid x_1, \ldots, x_{t+1}\big) \propto \mathrm{p}\big(x_{t+1} \mid z_t\big)\, \mathrm{p}\big(z_t \mid x_1, \ldots, x_t\big)\,,$$

so really what is going on here is importance sampling. This is also sometimes called the *smoothing* step.

2. **Filtering (updating/propagating)** Update the sufficient statistics with the new data point. For Bayesian inference, this means finding the updated sufficient statistics for the posterior given the new data point. The new particle states are drawn according to

$$z_{t+1}^i \sim \mathrm{p}\Big(z_{t+1} \mid z_t^i, x_{t+1}\Big)\,, \quad i = 1, \ldots, N_0.$$

This step is justified by the total probability expression

$$\mathrm{p}\big(z_{t+1} \mid x_1, \ldots, x_{t+1}\big) = \int \mathrm{p}\big(z_{t+1} \mid z_t, x_{t+1}\big)\, \mathrm{p}\big(\mathrm{d}z_t \mid x_1, \ldots, x_{t+1}\big)\,.$$

The integral is precisely taken with respect to the resample output of the previous step. This procedure is usually known as *filtering*.

3. **Learn**. Although this step need not be performed at every iteration, it is easy now to learn from the particles since the sufficient statistics assumptions imply

$$\mathrm{p}\big(\theta \mid x_1, \ldots, x_{t+1}\big) \approx \frac{1}{N_0} \sum_{i=1}^{N_0} \mathrm{p}\Big(\theta \mid z_{t+1}^i\Big)$$

and the posterior predictive marginal density

$$\mathrm{p}\big(x_{t+1} \mid x_1, \ldots, x_t\big) \approx \frac{1}{N_0} \sum_{i=1}^{N_0} \mathrm{p}\Big(x_{t+1} \mid z_{t+1}^i\Big)\,.$$

3.6. Particle filtering for the dynamic mixture model

Recall that our mixture models relies on an unseen component allocation k_t such that

$$\mathrm{p}\big(x_t \mid k_t = j\big) = N(x_t; \mu_j, \Sigma_j).$$

Let m_t be number of different observed components at time t. If we assume without loss of generality that when a new data point is from a component not observed yet, it will be assigned to the component $m_t + 1$. It follows that given our choose of prior, the posterior predictive has the form

$$\mathrm{p}\big(x_{t+1} \mid x_1, \ldots, x_t, k_1 = j_1, \ldots, k_t = j_t, k_{t+1} = j\big)$$

$$
\begin{aligned}
&= \mathrm{p}\big(x_{t+1} \mid z_t, k_{t+1} = j\big) \\
&= \begin{cases} St(x_{t+1}; a_{t,j}, D_{t,j}, \nu_{t,j}) & \text{if} \quad j = 1, \ldots, m_t \\ St(x_{t+1}; a_0, D_0, \nu_0) & \text{if} \quad j = m_t + 1 \end{cases}
\end{aligned} \tag{1}
$$

where $a_{t,j}, D_{t,j}, \nu_{t,j}$ and a_0, D_0, ν_0 arise from the choice of conjugate prior Normal Inverse Wishart, and can be derived analytically very easy due to the conjugacy. The details of their derivations are left to the reader. To obtain them, it is necessary to know the following sufficient statistics which are store in the essential state particle z_t:

- m_t the total number of components.
- $n_{t,j} = \sum_{s=1}^{t} \mathbb{1}(k_s = j)$ the counts per component.
- $\bar{x}_{t,j} = 1/n_{t,j} \sum_{s=1}^{t} \mathbb{1}(k_s = j) x_s$ the mean per component
- $S_{t,j} = \sum_{s=1}^{t} \mathbb{1}(k_s = j) x_s x_s^\top - n_{t,j}(\bar{x}_{t,j})^2$ the unscaled variance.

Now, instead of tracking the value of k_t into the essential state, we follow [7] and [5] and propagate it through the model using the particle filter. To do this, we start by noticing that

$$
\mathrm{p}\big(k_t \mid x_1, \ldots, x_t\big) = \int \mathrm{p}\big(k_t \mid z_t, x_t\big)\, \mathrm{p}\big(\mathrm{d}z_t \mid x_1, \ldots, x_t\big)\,.
$$

so we can also do an importance sampling step after the smoothing/reweighting step of the form, assigning $k_{t+1} \leftarrow j$ with probability

$$
\mathrm{p}\big(k_{t+1} = j \mid z_t, x_{t+1}\big) \propto \mathrm{p}\big(x_{t+1} \mid k_{t+1} = j, z_t\big)
$$

the probabilities of the right-hand side of the previous equation are obtained from (1). After assigning a component j to x_{t+1}, we proceed to update every particle from

$$
\mathrm{p}\big(z_{t+1} \mid k_{t+1} = j, x_{t+1}\big)\,,
$$

this takes the form of updating the number of clusters m_t if a new cluster is created, assigning an additional observation to $n_{t,j}$, and updating the mean and covariance of the selected component by adding x_{t+1} to it.

Putting all together, we get a predictive density

$$
\begin{aligned}
\mathrm{p}\big(x_{t+1} \mid x_1, \ldots, x_t\big) = \sum_{j=1}^{m_t} \omega_{j,t} St(x_{t+1}; a_{t,j}, D_{t,j}, \nu_{t,j}) \\
+ (1 - \textstyle\sum_{l<t} \omega_{j,l}) St(x_{t+1}; a_0, D_0, \nu_0).
\end{aligned}
$$

where the $\omega_{j,t}$ are obtained from the $BAR(G_0, \alpha_0, \rho_0)$ as detailed in previous sections. Further specific details of how to implement the particle filter can be found at [5]. We limit to say that it is also necessary to add to the essential state vector z_t information about the Beta innovations of the BAR generating process, as well as keeping track of the number of clusters that come from the previous period.

3.7. The scaling factor Λ

In [5] it is suggested to model Λ also as an autoregressive dynamic linear model. Due to time constrains, we simply ran a weighted expontential average. This is not the best choice but gave sensible results for demonstrations purposes, as shown in the next section. This form we obtain the full intensity of the Poisson model as $\lambda(x) = \Lambda f(x)$ with f the density obtained from the dynamic mixture model.

4. Model Implementation

This section presents the details of the model implementation.

4.1. R-Package

We implemented the model as an R package called "pldensity". The detailed code can be found in `https://github.com/BigDataStats/pldensity/`. The developed tool uses `RcppArmadillo`. The user is able to download the package from our GitHub page using the command:

```
devtools::install_github("BigDataStats/pldensity")
```

The developed package provides the following main functions:

- `ddpn_init`
  This function initiates the hyper parameters of the prior.
  Input: number of particles and hyperparameters $\alpha, \lambda, \kappa, \nu, \Omega, \rho$.
  Output: model (object of class DDPN).

- `ddpn_mix`
  Uses the dynamic particle Dirichlet process filter algorithm to train the model. It obtains the weights, center and covariance of each mixture component.
  Input: model (object of class DDPN).
  Output: trained model (object of class DDPN).

- `ddpn_eval`
  This function receives the trained model and new points and evaluates the density of those new points.
  Input: trained model (object of class DDPN), x (new points in matrix format).
  Output: final model (object of class DDPN).

- `ddpn_marginal`
  Obtaines the marginal DDPN in a set of specified dimensions Input: DDPN trained object and integer index vector
  Output: DDPN of smaller dimension on the selected index

- `ddpn_conditional`
  Evaluates conditional densities from the joint trained DDPN model.
  Input: DDPN object trained object, an index of dimensions in which to evaluate, a matrix of new values in which to evaluate, and index of dimensions in which to condition, a matrix of values in which to condition
  Output: A matrix where row represent points in which to evaluate, and columns the density condition on each row of the conditioning values matrix

- `spatial_plot`
  Uses the final trained model and points to evaluate the density and presents the results in a Leaflet Map. Optionally, it also evaluates a grid in the map and creates a heat map.
  Input: final model (object of class DDPN), x (new points in matrix format).
  Output: map plot.

4.2. Cross-Validation

A cross-validation framework is implemented in order to optimize the parameters for the density estimation. More details can be found on `https://bigdatastats.github.io/pldensity/htmldocs/crossvalidation.html`.

4.3. Model Limitations

The main limitations of the method used are:

- Bayesian methods depend on simulations which make them slower than Frequentist methods of point estimation.

- The current implementation does not take information from previous weeks at the same time, but only from the previous time period.

4.4. To-do list

- Implement the spatial regression for the interesting quantities: for example the travel distance, time, or cost, depending on the origin or destination. We plan to use graph-fused lasso for this.

- We are not exploiting that we have a Bayesian method to assess the uncertainty of the model.

- We use a simple weighted average model for Λ the total scaling factor. It'd be better to implement also a dynamic model to properly filter out the noise.

5. Ride-Sourcing Demand

We applied the developed framework to model demand intensity to the ride-sourcing dataset with the goal of providing hourly demand intensity maps.

5.1. Density and Intensity Output

In this section we present an example of the output using 15 minutes intervals for trips during Wednesday, March 1st, 2017. The demand density and intensity are show in Figures 3 and 4, respectively. We present an animation of the results for the entire day in our GitHub at

`github.com/BigDataStats/pldensity/blob/master/htmldocs/Intensity-c.gif`
`github.com/BigDataStats/pldensity/blob/master/htmldocs/Density-c.gif`

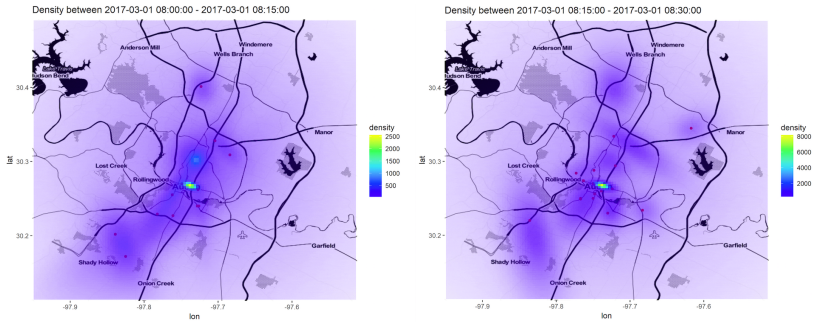
As explained in the previous section, the density f is obtained from the dynamic Dirichlet process mixture and scaling factor Λ from a simple exponentially weighted average so that the intensity is obtained $\lambda(x) = \Lambda f(x)$.

5.2. Analysis of Demand Example

The example of the application consists of mapping the demand density during weekday and weekends. We selected hourly intervals during two different hours, 8 AM and 2 AM, to compare the behavior of short and long trips during recreational and commuting trips. We classified the short trips as those with a trip longitude shorter or equal than 6 km, and long trips with the distance greater than 6 km.

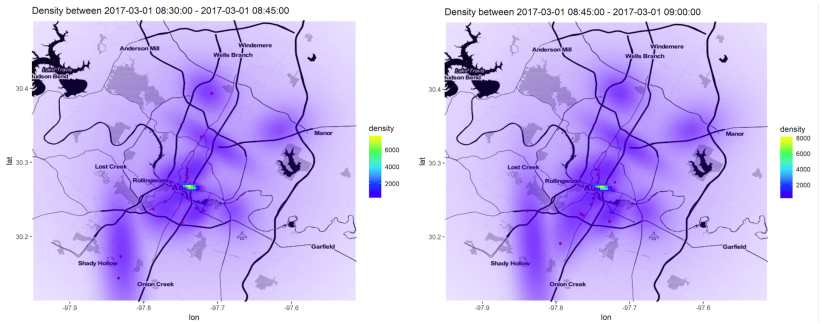
The demand density for weekday is shown in Figure 5, while the weekend is shown in Figure 6. We can observe that the separation per trip longitude provides valuable information. During the weekday early trips (2 AM), the short trips are concentrated on the downtown and nearby areas, while the majority of long trips are requested from the airport. During weekends the early morning long trips seem to be more concentrated in the Domain area, a transit-oriented development (TOD) with high-density of offices, retails, restaurants, and residential center.

The demand density for weekday commuter trips peak hour (8 AM) show that the short trips concentration is reduced compared to early morning (2 AM) and presented a marked north and south distribution. While long trips are spread around the central area and an isolated cluster is found for the Domain region. The weekend trips for 8 AM present a different travel patron. We can observe that short trips cover a wider area in the central region. Long trips present four different concentration areas, downtown, the airport, the Domain, and south Austin.



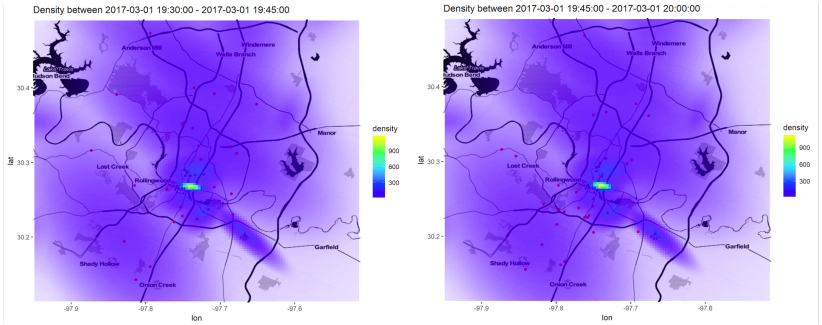
(a) Trips from 8:00 to 8:15

(b) Trips from 8:15 to 8:30



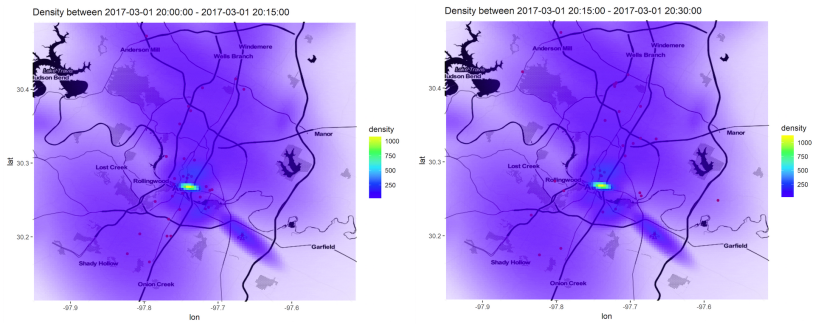
(c) Trips from 8:30 to 8:45

(d) Trips from 8:45 to 9:00



(e) Trips from 19:30 to 19:45

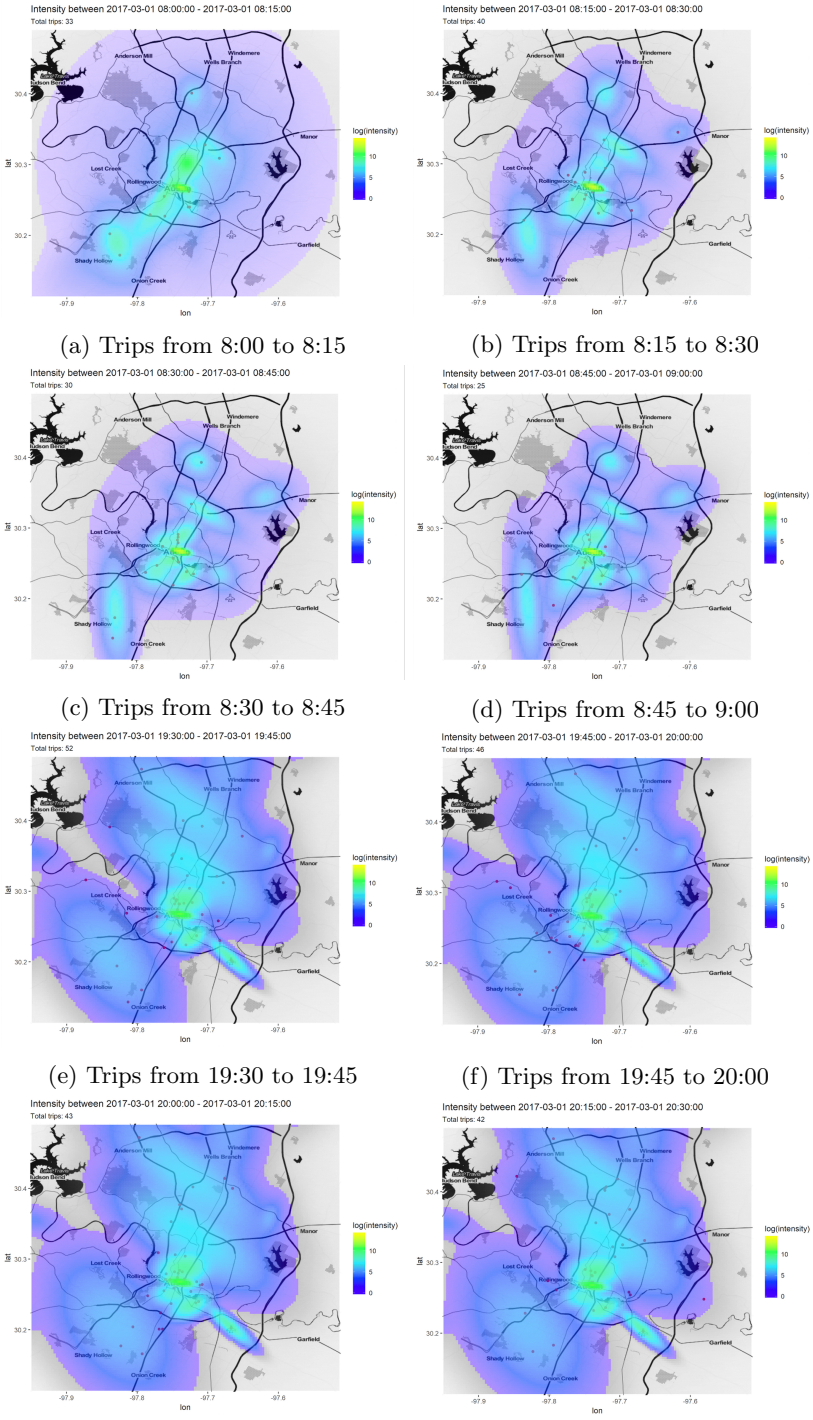
(f) Trips from 19:45 to 20:00



(g) Trips from 20:00 to 20:15

(h) Trips from 20:15 to 20:30

Figure 3: Demand density for trips during Wednesday, March 1st, 2017



(a) Trips from 8:00 to 8:15

(b) Trips from 8:15 to 8:30

(c) Trips from 8:30 to 8:45

(d) Trips from 8:45 to 9:00

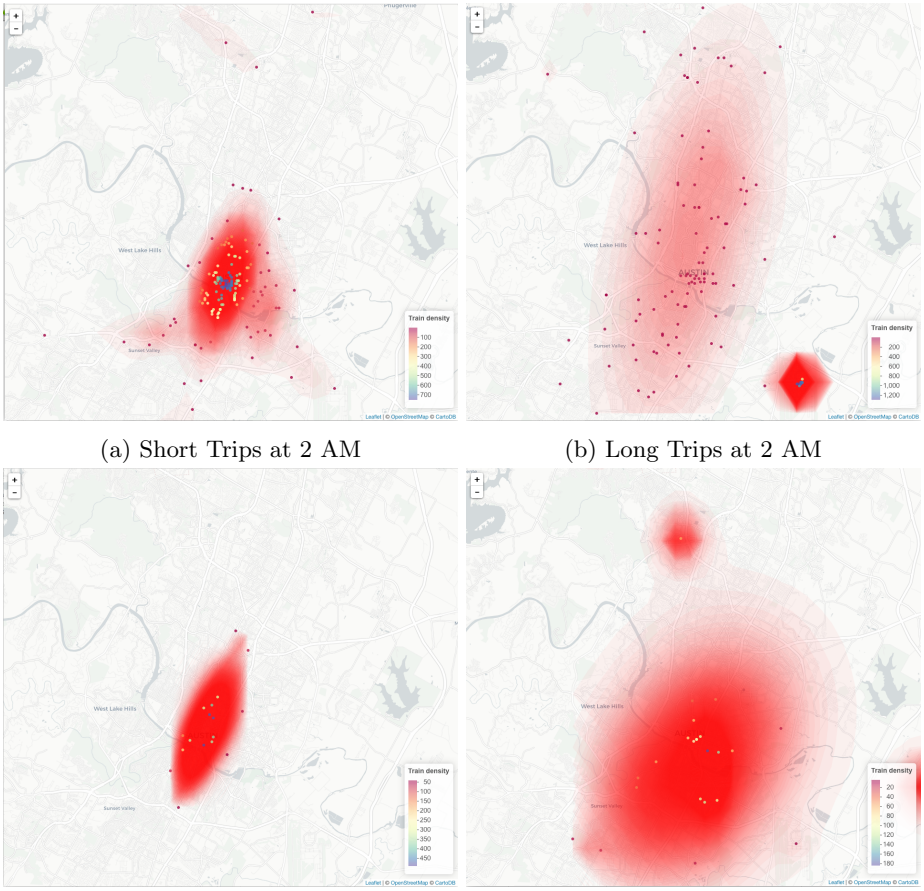
(e) Trips from 19:30 to 19:45

(f) Trips from 19:45 to 20:00

(g) Trips from 20:00 to 20:15

(h) Trips from 20:15 to 20:30

Figure 4: Demand intensity for trips during Wednesday, March 1st, 2017



(a) Short Trips at 2 AM

(b) Long Trips at 2 AM

(c) Short Trips at 8 AM

(d) Long Trips at 8 AM

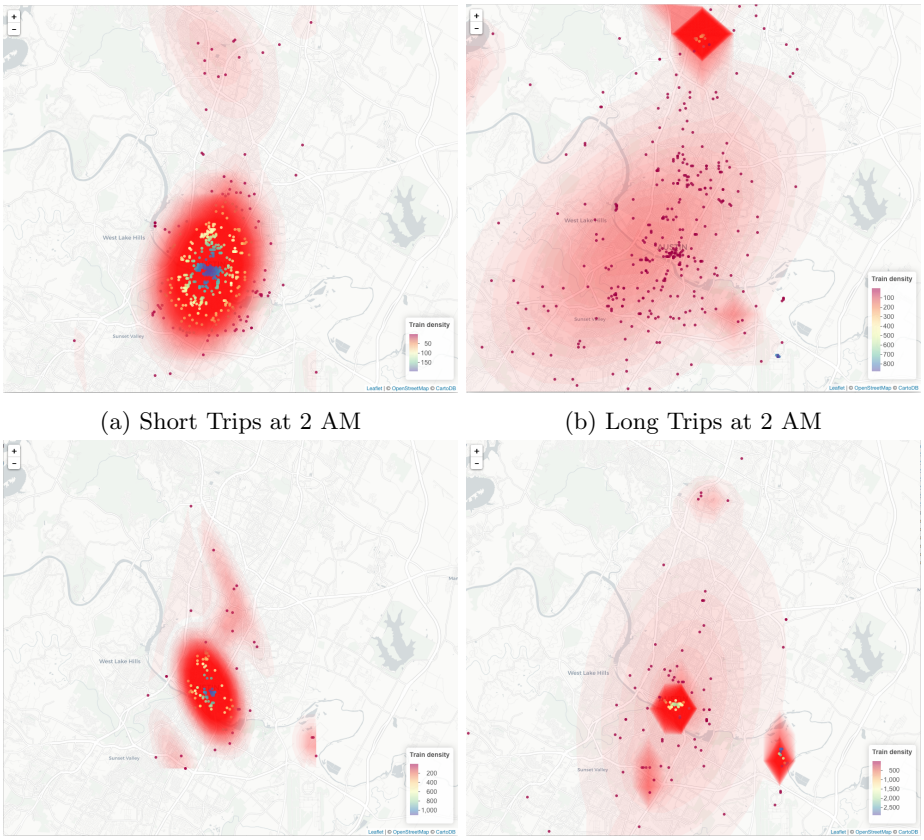
Figure 5: Origin Demand Density Weekday (Wednesday April 5th, 2017)

6. Conclusions

This research study implemented a methodological framework to track demand intensity of ride-sourcing trips. We also provided a tool that the user can utilize to evaluate intensity using their own dataset. The main advantage of the method is that it allows the user to track the demand intensity spatially, which can have several applications for transportation agencies, city authorities, and ride-sourcing companies.

References

[1] S. Shaheen, A. Cohen, and I. Zohdy, "Shared mobility: Current practices and guiding principles," tech. rep., 2016.



(a) Short Trips at 2 AM

(b) Long Trips at 2 AM

(c) Short Trips at 8 AM

(d) Long Trips at 8 AM

Figure 6: Origin Demand Density Weekend (Saturday April 8th, 2017)

[2] B. Carson, "Lyft tripled its rides in 2016," *Business Insider*, Jan 2017.

[3] R. Holt, A. Macdonald, and P.-D. Gore-Coty, "5 billion trips," *Uber Newsroom*, Jun 2017.

[4] A. Samuels, "Uber, lyft returning to austin on monday," *Texas Tribune*, May 2017.

[5] M. A. Taddy, "Autoregressive mixture models for dynamic spatial poisson processes: Application to tracking intensity of violent crime," *Journal of the American Statistical Association*, vol. 105, no. 492, pp. 1403–1417, 2010.

[6] A. Ishwaran and M. Zarepour, "Markov chain monte carlo in approximate dirichlet and beta two-parameter process hierarchical models," *Biometrika*, vol. 87, pp. 371–390, 2000.

[7] C. M. Carvalho, H. F. Lopes, N. G. Polson, M. A. Taddy, *et al.*, "Particle learning for general mixtures," *Bayesian Analysis*, vol. 5, no. 4, pp. 709–740, 2010.

A. Database Description

Table A.1: Variables Description

Trip and Rider Information			
ride_id	Unique rider identification (anonymized)	**start_location_long**	
rider_rating	Star rating assigned to rider	**start_location_lat**	
started_on	Date / time the ride initiated	**end_location_long**	
created_date	Date / time this data was created	**end_location_lat**	
updated_date	Date / time this data was updated	**distance_travelled**	Measured in meters
completed_on	Date / time the ride was completed	**status**	Dispatch status
estimtated_time_arrive	Date / time the ride was completed		
Driver and Car Information			
driver_id	Unique driver identification (anonymized)	**driver_accepted_on**	
active_driver_id	Unique active driver identification (anonymized)	**car_id**	Unique car identification
driver_rating	Star rating assigned to driver	**color**	
driving_time_to_rider		**make**	
dispatch_location_lat		**model**	
dispatch_location_long		**year**	
driving_distance_to_rider	Measured in meters	**car_categories_bitmask**	
dispatched_on		**rating**	Average star rating at the time of dispatch
driver_reached_on	Date / time driver arrived at the passenger location	**requested_car_category**	
Payment and Cost Information			
charity_id	Unique charity identification	**total_fare**	Total Cost -Tip -Roundup -Processing Fee
free_credit_used	Promotional credit used	**rate_per_mile**	
surge_factor	Factor by which the fare is multiplied	**rate_per_minute**	
round_up_amount	Amount of $ donated to charity	**time_fare**	Amount charged for time ($0.25/min)
promocode_redemption_id	Unique promocode redemption id	**tipped_on**	
base_fare		**tip**	Amount of $ tipped in app
Weather Information			
HourlyVisibility	**HourlyWindDirection**	**DailyDeptNormalAverageTemp**	**DailyPrecip**
HourlyDryBulbTempC	**HourlyPrecip**	**DailyAverageRelativeHumidity**	**DailyAverageWindSpeed**
HourlyRelativeHumidity	**DailyMaximumDryBulbTemp**	**DailySunrise**	**DailyPeakWindSpeed**
HourlyWindSpeed	**DailyMinimumDryBulbTemp**	**DailySunset**	