Optimal Bike Allocation for ValleyBike Share Program

Brian Dang bldang
Guilherme Silva guilhermesil
Nilay Sadavarte nsadavarte
Ben Silver bcsilver

1 Introduction

Bicycle-sharing systems, or bike-share systems, are systems set up within various locations in order to offer the availability of bikes to be used by people to get from one place to another. Typically, stations are positioned at popular locations in order for the users of the system to conveniently pick up and drop off their borrowed bikes. While these systems offer an excellent service by providing an important and useful mode of transportation in beneficial locations, problems such as asymmetric demand and rebalancing are challenges that prevent bike-share systems from being more successful. These terms and our approach to resolve them will be discussed further in this paper.

2 ValleyBike

Western Massachusetts is home to a region known as The Pioneer Valley. Within the valley is the Franklin, Hampshire, and Hampden Counties. This area contains buzzing communities in Springfield, Holyoke, Northampton, Hadley, and Amherst. As of 2018, these communities are tied together by another means: the newly created ValleyBike Share. This recently established bike-share system has added over 50 bike stations where customers can take out, ride, and return their borrowed bikes. Offering electric pedal assist bikes, the ValleyBike Share service delivers a new means for public transit.

With over 126,000 routes already taken, this certainly is a useful and popular service. While boasting a significant initial year of bike usage, ValleyBike is not immune to the challenges that bike-share systems face. Having too many or too few bikes at a station at any given time is one way for potential dissatisfied customers.

3 Our Goal

We are interested in an optimal allocation of bikes at each station such that the number of stock outs are minimized for each station. We define a stock out as an instance in which a customer would either not be able to dock their bike as a product of all docks being occupied or not be able to take a bike because there are none available.

4 Research

The first paper we read was "Data-driven rebalancing methods for bike-share systems" (Freund et al., 2017). From this paper, we learned about the nature of bike share systems, and its' unique "asymmetric demand" problem. Bike stations spread out across a region, have customers demanding the service of the bike stations availability. This demand is seeking both bikes and open docks (to return their bike). When a customer is unsatisfied, it is due to a stockout – which means there is either no available bikes or no available open docks.

Rebalancing is the term for adjusting the number of bikes at any given station in order to reset its bike and dock values to better serve the customer demand. This paper discussed different methods of rebalancing both during the day and throughout the night.

The paper also discussed a user dissatisfaction function. This UDF defines the process of removing and docking bikes where removing reduces the total number of bikes at a station by one and docking increases the total number of bikes by one. Unless, of course, if the station is full then a bike cannot be docked and if the station is empty a bike cannot be removed.

The paper also mentions its' use of the Poisson process which is something we were able to incorporate into our model, and we will discuss later.

The next paper we read was "Minimizing Multimodular Functions and Allocating Capacity in Bike-Sharing Systems" (Freund et al., 2016). In this paper, we read about how a greedy algorithm takes out all the bikes and then adds them back in one by one. By then improving the objective function little by little, the allocation of bikes can be adjusted until no more room for improvement exists.

Another paper we read was "Data Analysis and Optimization for (Citi)Bike Sharing" (O'Mahony and Shmoys, 2015). This paper mentions applying penalties to when a station doesn't have ideal values. That is, too many or too few bikes. Also, this paper suggests "minimizing the sum of the differences..." which is another concept we were able to incorporate into our model.

Since our group's main objective was to find the appropriate number of bikes/docks for each station at any given time, we took various pieces from these papers and developed our models.

5 Initial Work and Data Collection

We highlighted some of the data that would help us with the solution to our problem. This included: the total number of bikes and the number of stations, their locations, capacities, and the rates of removal and docking of bikes.

We approached the problem by treating it as a minimization problem, trying to minimize the difference between the number of bikes we want to allocate to each station and the number we would actually be able to supply.

We wanted to incorporate certain penalty values depending on the conditions that each station was in. Our goal was to penalize not having the optimal allocation at a station, but to varying degrees considering the current rates at the station and whether we were allocating more or less bikes than desired, which led us to having four possible scenarios.

We used variable $S_{st} = |x_{st} - y_{st}|$ to model our difference, but in order to take the situation into consideration we expanded on S_{st} to become the following:

$$S_{st} = x_{st} - y_{st}, r \geq d \tag{1}$$

$$S_{st} = y_{st} - x_{st}, r \leq d \tag{2}$$

$$S_{st} = x_{st} - y_{st}, r \leq d \tag{3}$$

$$S_{st} = y_{st} - x_{st}, r \geq d \tag{4}$$

Where scenario 1 meant that there were more bikes allocated than optimal and the rate of removal from that station was higher than the rate of docking. Scenario 2 means that there were less bikes allocated than optimal and the rate of removal was lower than the rate of docking. Scenario 3 means that there were more bikes allocated than optimal and the rate of removal was less than the rate of docking. Scenario 4 means that there were less bikes allocated than optimal and the rate of removal was higher than the rate of docking.

We further constrained S_{st} such that we grouped similar situations together and came up with the following:

$$S_{st} \geq 0$$

Enforces that S_{st} is a nonnegative number.

$$S_{st} \leq x_{st} - y_{st} + B_2 * C_1 + B_4 * C_2$$

$$S_{st} \geq x_{st} - y_{st}$$

If either scenarios B_2 or B_4 are activated, it is scaled by an associated constant C such that it compensates for the discrepancy and the second part makes it an equality.

$$S_{st} \leq y_{st} - x_{st} + B_1 * C_3 + B_3 * C_4$$

$$S_{st} \geq y_{st} - x_{st}$$

Similar to the previous set of equations, if either scenarios B_1 or B_3 are activated, it is scaled by an associated constant C such that it compensates for the discrepancy and the second part makes it an equality.

In order to know how to penalize our model, we created a new variable w_{st} such that it takes on the value of the penalty multiplied by the value of S_{st} which is determined by the scenario it falls under and can be described by the following:

$$w_{st} = \begin{cases} P_{small} * S_{st} & S_{st} = x_{st} - y_{st}, r \geq d \\ P_{small} * S_{st} & S_{st} = y_{st} - x_{st}, r \leq d \\ P_{large} * S_{st} & S_{st} = x_{st} - y_{st}, r \leq d \\ P_{large} * S_{st} & S_{st} = y_{st} - x_{st}, r \geq d \end{cases}$$

Our motivation behind assigning either a small or large penalty was determined if the scenario was worse for the model. Two of the situations are

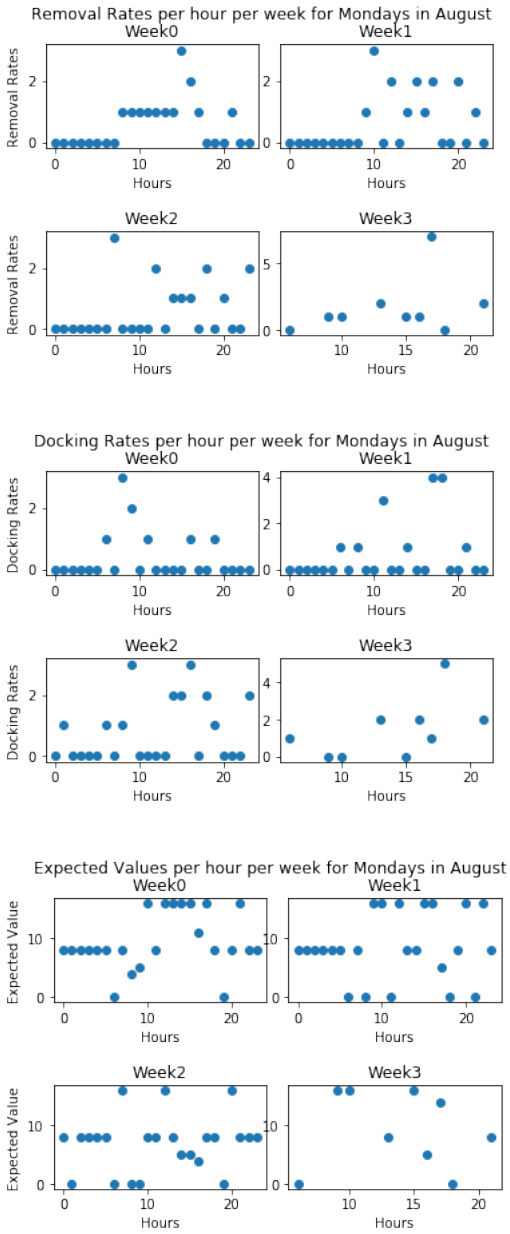
worse than the other two; for example, having more bikes than ideal at a time where more people are removing bikes than returning is better than having more bikes than ideal and a higher rate of returning bikes (that would only increase the delta – difference between the actual number of bikes and the ideal number). Likewise, there are situations that consider the reverse ratios.

We also decided on implicit constraints. This included requiring all number of bikes to be non-negative integers, the number of bikes at any station can never exceed the capacity of the station, the number of total bikes in the system is a constant, etc.

The data provided by ValleyBike allowed us to calculate the rates of removal and docking for each station within a given time period. The 2019 data was relatively easy to work with as it provided each route with its start and end date and time of day as well as stations. Using Python and the Pandas library, we formatted the csv files into a dataframe containing all of the relevant information for the rides. From this initial dataframe, we branched it off into two separate dataframes that were grouped by starting station and ending station, respectively. In each dataframe we further grouped the entries by the week of the month, the day of the week, and the hours of that day and extracted the rates of docking and removal by looking at the number of elements within each group.

Regarding the 2018 data, it was not formatted as nicely as the 2019 and therefore some more work went into getting the rates for that dataset, which included matching gps locations and calculating the time a ride started and ended. Eventually the data was fully collected and we averaged out the 2018 and 2019 rates with the corresponding days.

When vizualing rates for the stations, we came across a problem, which was that the rates were very low or, for large ranges of time, simply zero. This made it so our expected value, which was calculated by seeing if the rate at that station at that time was within its capacity and if it was above capacity, we would take the ratio of the rates and allocate according to that ratio. With the patches of the rate being zero, we decided to simply say it can be split in half, or can just be interpreted as just leaving it alone for that hour.



This initial model proved to be uninformative and rather inefficient, as the the constantly changing rates would suggest for frequent rebalancing but the rates were always low enough that there was not always an urgent need to rebalance every hour. Given this inefficient nature, we decided that hourly rebalancing was not ideal and we should limit rebalancing to once or twice a day and simply take the busy hours of the day into consideration for the rate, so our collected and calculated data was still essential, it just needed to be further tweaked.

6 Final Model

Our current model includes these features: an iterative approach by adding one bike at a time to each station; each station is chosen by what will mini-

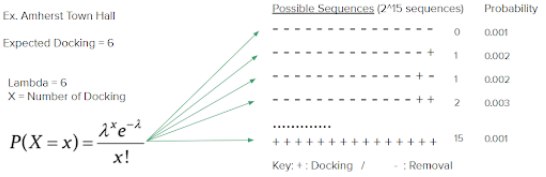
mize the number of stockouts through a greedy algorithm. The steps involved in our algorithm are:

1. Gather data on docking rates for each location at a given time
2. Use Poisson distribution to give a probability to a sequence of docking and removal
3. Calculate the number of stockouts with the current allocation of bikes
4. Get the weighted number of stockouts through step 2 and 3
5. Temporarily add one bike to each station to see what decreases the number the most
6. Repeat until the loss does not decrease anymore or there are no more bikes

For our data, we compared the number of bikes being docked at a station with the number of bikes being removed at that station within that given time frame. This ratio is docking to removing. The ratio produces the rate for that time frame (i.e. 4 docking and 4 removing gives a ratio of 4:4 and a rate of .5). The docking rate will become the expected number of bikes that should be docked at the station.

Station	Ratio (Docking:Removal)	Rate	Capacity	Rate * Capacity = Expected # of Docking
Amherst Town Hall	4:4	0.5	12	6
UMass Haigis Mall	2:6	0.25	16	4
UMass Knowlton	9:0	1	10	10
University Drive	2:8	0.2	10	2
...	...	...	...	...

In order to come up with the number of possible stockouts, we created 32,768 sequences (2^{15}). Each possible sequence represents an arrangement of 15 customers approaching a station looking to either dock or remove a bike (+ indicates docking, - indicates removal). 32,768 is all possible situations from all 15 looking to remove bikes, to all 15 looking to dock bikes, and every combination in between. A Poisson distribution applied to these sequences determines the probability of likelihood of each situation and determines which is the most likely.



Given the current allocation of bikes at a given station, we can then calculate the number of stockouts. While we begin by setting the number of bikes at any given station to 2 in order to run through these sequences, we add in bikes one by one at each iteration in order to minimize the average number of stockouts at every station. After calculating the number of stockouts with a given allocation for every sequence, a Poisson distribution is used with λ being the average number of docking at a station and x being the number of docking in a sequence. The weighted number of stockouts is the product of the probability the sequence can occur and the number of stockouts associated with it. Afterwards, we take the sum of it to be the loss of that station. This is done for every station.

After the loss is calculated between each station, we add one to their allocation and calculate the loss. Between all the stations, the maximum difference in loss will be the station in which a bike will be added. Once the bike is added, the iterative step starts over from the top with the new allocation as input. The process is repeated until there are no bikes to add or any bike added to the system will produce a suboptimal allocation.

7 Results

The final model was used to calculate the optimal allocations for each station at a certain time period and in this case, it is a Monday in August. In Figure 1, the station Amherst Town Hall would be optimal with 8 bikes and the Basketball Hall of Fame with 8 bikes. For some of the data like UMass Southwest with 10 bikes, it makes sense to have a high number of bikes when considering its capacity. This is due to the fact that the station is very active when it comes to removing bikes. This happens to be reflected in our model.

We initially set the number of bikes that can be used by ValleyBike to be 500. After the model had finished running, it did not use all of the 500 bikes. In fact, it was optimal with 72 bikes remaining or 428 bikes in the system. Intuitively, having more bikes in the system should produce better results, but in this case, having more bikes can lead to more stockouts. From a rider's perspective, there might not be enough docks when they take a bike out from a station to go to another.

The final loss on the system is around 239.80 when summed across all the stations. Even though

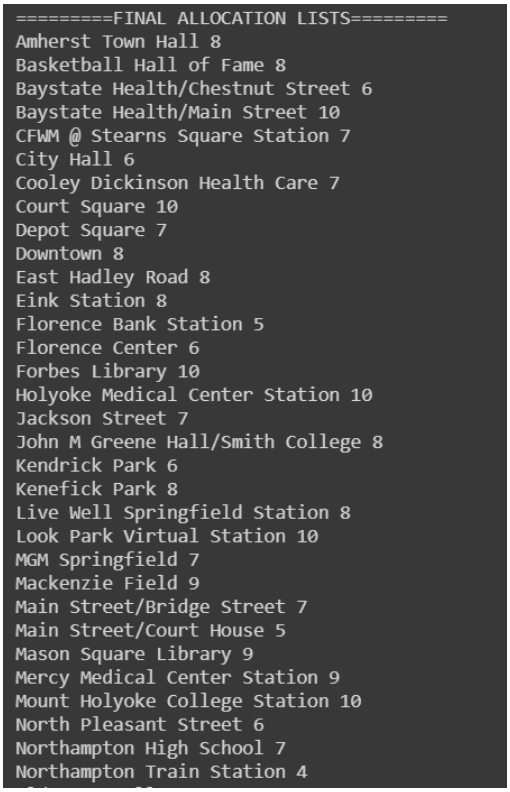


Figure 1: Example of a Final Allocation List on a Monday in August

this does seem a little high, it has to be considered that the result came from all 57 locations on 32,768 sequences.

8 Conclusion

Using our final model with weighted probabilities of sequences in order to predict the allocation of bikes that would minimize the average number of stockouts proved to be much more informative and effective than our initial model. Generalizing and averaging rates out daily instead of hourly also improved our results as there weren't cases of rates of zero any more and every station had a concrete allocation rather than an estimation given by the ratio of rates. The final model was intensive with calculations and took around a minute to fully run for each month, but it produced values that we feel are worth that run time. Given our model, we can use the trends of the past two years in order to predict what they will be in the coming year and as time passes and we accrue more data, the model will only get more precise, as we are currently just taking the average of two data points for each day of the week in the month.

Our ideal rebalancing schedule would be twice a day, once during the middle of the busy 12 hour period of the day and once again at the end of the 12 hours. This rebalancing schedule provides enough bikes to satisfy the needs of the night and early morning period, as most of it was zero or one bike per hour, and the mid-day rebalancing allows ValleyBike to restock at stations that are in need and finish the day out.

As far as suggestions for ValleyBike, we suggest that they implement a system that keeps real-time statistics on their bikes and stations, including being able to easily access the number of open docks and available bikes at a given station. This real-time tracker would allow them to have a system in place that can rank stations on their need of bikes or open docks and alert them for more efficient rebalancing. A helpful feature that could also be implemented is a tracker that counts how many people have clicked on a station in the app to either check if it has any available bikes or docks. With this system in place, you would be able to have a rough estimate of the demand for that station, because if the docks were completely empty or full, you would not know if there would be a stockout event because there is no way to gauge whether or not someone wants to remove or dock a bike.

Overall we hope that you find our model helpful to be able to find optimal allocations for each station on a given day of a given month and that it may reduce the number of stockout events.

References

Freund, D., Henderson, S. G., and Shmoys, D. B. (2016). Minimizing multimodular functions and allocating capacity in bike-sharing systems.

Freund, D., Norouzi-Fard, A., Paul, A., Henderson, S., and Shmoys, D. (2017). Data-driven rebalancing methods for bike-share systems.

O'Mahony, E. and Shmoys, D. B. (2015). Data analysis and optimization for (citi)bike sharing. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, pages 687–694. AAAI Press.