Course Project

0. Introduction

In this paper, I will be taking a look at the project Increasing Bike Share Efficiency on the *Informs* website. As the name describes, this project tackles the problem of optimizing the distributions of bikes in a bike sharing system in order to increase the efficiency of the system.

1. Bike Sharing and The Bike Rebalancing Problem

Bike sharing systems are public transportation systems adopted by many cities to reduce traffic congestion and pollution and provide a more sustainable means of transport. They have become a popular transportation option in many cities because they are a green and convenient method of transportation, and act as an essential feeder for other transportation modes such as buses or trains.

In general, there are two different categories of bike sharing systems: dock-based (or station-based) systems and free-floating services. In line with the original project being discussed in this paper, we will concentrate our analyses on dock-based systems only. What dock-based systems entail is this: they require users to pick up and drop of bikes at docking stations which are located specifically within the boundaries of a city. Furthermore, these docking stations are outfitted with a set number of docks.

What this means is that the capacity of each station is limited to the number of docks available at that station. Upon completing a trip and arriving at their destination, a user may find that all the docks at the station are full, requiring the user to travel further to another nearby station to drop off the bike and walk back to the intended destination, incurring higher costs and consuming more time. Or, station at which the trip starts might be empty, forcing the user to walk to another station or use another means of transport. Thus, dock-based bike sharing system rely on rebalancing operations to maintain the usability of such systems and limit the inabilities to use a station, either for the pick-up or drop-off of bicycles.

The Bike Rebalancing Problem consists of using a set of vans to move bicycles across different stations for rebalancing purposes. Static rebalancing strategies are performed when the system is offline or when the system is barely used, i.e., during night time. On the other hand, dynamic rebalancing strategies consists of moving bicycles across the stations when the system is operating and is used by the users. Dynamic rebalancing operations need to be performed in a short amount of time to avoid large amount of bicycles being unavailable to users due to operators moving them across different stations. For this reason, bicycles are usually moved among nearby stations (from full to empty ones). However, bike rebalancing by vehicles is costly and not always effective. More recently, different pricing schemes have been proposed to engage users to help reposition the bikes, as a type of rebalancing called volunteer or crowd sourced rebalancing.

Human mobility patterns in time and space unavoidably lead to a bike imbalance problem. Over the years, many different approaches have been developed to solve this problem - from optimizing rebalancing routes to station locations to fleet size - and the project we are going to talk about is just one more of such studies.

2. Discussing The Original Solution

Referring back to the project, we find that the researchers involved in this project developed a novel solution. Their solution is in two parts:

1. They opted to optimize the allocation of docks to stations, and
2. Employed an incentive scheme to crowd source rebalancing.

The article did not go into detail on the approach used by the researchers on this project, nor is it the purpose of this paper to discuss the researchers' approach. Instead, I will be discussing the analytics models that I believe would have been used to develop the solution described in the article.

As the title of the article suggests, the goal of this analysis is to increase the efficiency of bike sharing systems. This will be discussed through the lens of the original solution, which is to allocate docks to stations, and not through traditional rebalancing efforts which involves moving the bikes instead. The focus on the allocation of docks also shows that this solution, in contrast to traditional rebalancing efforts, is intended as a long-term solution, possibly over years, since obviously docks are not so easily transported around as bikes are.

Finally, there are some things that this analysis will not address. Optimizing the location of stations is not within this paper's scope, nor is optimizing the fleet size (number of bikes in the system). Improving the efficiency of bike share systems is a complex problem, and we will only concern ourselves with one part of it – optimizing the distribution of docks at each station in order to stay true to the solution in the article.

In this paper, I have proposed a methodology to tackle the optimization problem of the allocation of docks. The analytics models discussed will be building up to the final optimization model. Before we can get there, we will first need two other models: a time-series forecasting model to predict demand at each station, and an application of the Louvain algorithm to find communities of interconnected stations. Then we will apply the results of these two models to a queuing optimization model to find the best way to allocate docks.

After that, we will also briefly discuss the process of finding the best way to incentivize crowd sourced rebalancing with a multi-armed bandit experiment.

Throughout the paper, we will be taking a closer look at the analytics model chosen, as well as the advantages and disadvantages of each model

3. Time-Series Forecasting: Predicting Demand At Stations

Before we can do anything, it is essential to predict the supply and demand of bikes at stations. The prediction of demand for shared bikes is important for planning the allocation of docks because we can only reach the optimal allocation strategy when the predicted demand reflects the actual needs of the users.

To do this, we will be analysing time-series data of bike trips, using a time-series model, to predict the demand at stations.

Here are some types of data that should be considered for this model:

- Start and Endpoint Data: Start and endpoint data includes data such as the starting or ending coordinates of the bike trip, station, time and date.
- Subscription or Membership type: Bike sharing companies often offer subscriptions to users that provide benefits such as discounted rates for certain distances, or times of the day. User behavior differs based on the presence and type of subscription.

These types of data should be fairly readily available for analysis as it would come from logs of bike trips that bike sharing companies record every time a user rents a bike. However, these are automatically logged data by machines and we should expect missing data or outliers in cases where the machine fails. Also, public bikes can be involved in accidents or thefts which may also interfere with accurate data collection.

After data is collected, we will want to use a time-series model to forecast the demand of bikes at bike stations. The demand of bikes can be described by the number of bikes available at each station and waiting times. These variables are observations and not variance. Of the time-series model we have learnt in this

course, GARCH is no longer an option and so this leaves us with two types of models to consider: exponential smoothing or ARIMA.

Bike demand at bike sharing stations is non-stationary data, that is, the demand follows trends and seasonality. In general, exponential smoothing is better at handling non-stationary data while ARIMA forecasts based on stationary data. Given this, exponential smoothing seems like the default choice for non-stationary data. However, remember that the optimization of dock allocation is more concerned with the long-term. For longer-term forecasts, forecasting on stationary data is more reliable than forecasting on non-stationary and noisy data, so the ARIMA model is preferred.

Further, the non-stationary nature of bike demand will not be an issue for the ARIMA model. ARIMA can transform non-stationary data into stationary data using differencing (that is, it models the differences between observations which are stationary, rather than the non-stationary observations itself) and thus can also work on non-stationary data.

As a note, it is possible to test if data is stationary or non-stationary. Even if it turns out that the data is stationary, we can still use an ARIMA(p,0,q) model where the order of differencing is 0 and no differencing is done. The value of the ARIMA model over exponential smoothing in this project mainly comes from ARIMA's better long-term forecasting.

For added certainty, it's not an uncommon practice to try several time-series models before performing cross-validation on them to determine the best model using AIC or BIC as metrics.

Since time-series forecasting predicts the future based on past values, most time-series models do not do very well at forecasting very long-term time-series. The further into the future we are trying to predict, the larger the error terms will be. Even though ARIMA is better than exponential smoothing at long-term forecasting, it still has this pitfall. With the understanding that the allocation of docks is meant to be a somewhat longer-term solution, we can try re-running the model every six months to year or so to monitor the demand of bikes since the last time.

Alternatively, we can build a more complex model by combining ARIMA with boosting methods or neural networks in order to forecast further into the future.

4. Communities in Graphs: Identifying Interconnected Stations

If we consider every station as a node, and travel routes between stations as the edges, we can construct a transportation network of the bike sharing system in a city to understand human mobility patterns. Using the Louvain algorithm on this network, we should be able to identify communities of interconnected stations.

Interconnected communities in this case are groups of bike sharing stations that people frequently travel between. For example, a bike sharing station at an office building and a bike sharing station at a nearby train station would be considered interconnected stations; people who work at the office building will travel between these two bike-sharing stations daily. In the morning, there would be a high demand of bikes from the train station and a high supply of bikes at the office building. In the evening, it would be the opposite. Traffic between interconnected stations is related: an increase in supply of bikes to one interconnected station is due to the increase of demand of bikes at another interconnected station.

Thus it stands to reason that if we increase the number of docks at one of these interconnected stations, we should also increase the number of docks at the other interconnected stations.

To find the interconnected stations, we will need the following data:

- Locations of stations recorded in coordinates
- Approximate travel time between stations. This can be calculated using in-house bike trip data (eg. using start/end time and stations like what was used in the previous section) or using apps like GPS or Google Maps.

- Frequency of trips between stations which can be derived by tracking how each bike ID moves between stations

Both the travel time and the frequency of trips will contribute to the weight of each edge.

The main downside of the Louvain algorithm is that it's not very good at mapping overlapping communities. It produces only non-overlapping communities, which means that each node can belong to at most one community. This is usually not representative of real-world situations. For example, bikes may be arriving to the office building from not one but two train stations, but not between the two train stations themselves. These are two different train-to-office-building communities that have overlapped on the office building. One possible algorithm that does better with overlapping communities is the Leiden algorithm but has not been taught in class.

5. Queue Optimization: Optimizing Allocation of Docks At Stations

Having modeled bike demand using an ARIMA model and human mobility patterns with the Louvain algorithm, we can move on to optimize the allocation of docks according to queuing theory.

Like any other queueing model, we have:

- The arrival rate of customers (bike demand),
- The service rate (bike supply),
- The service cost (cost to install or maintain docks), and
- Customer waiting time for a bike or dock.

Our queue optimization model aims to minimize the total cost of service and waiting by finding the balance between bike demand and bike supply at stations. On one hand, increase docks at stations increases the capacity of the station to hold more bikes, which leads to a lower waiting time for bikes or docks. On the other hand, increasing docks comes at a service cost to the company, and installing too many docks will be a waste.

Here are some types of data, or variables, to be considered for this optimization problem:

- Interconnected stations: This is the output from the Louvain algorithm in the previous section.
- Expected traffic at each station: This is the output from the ARIMA model in the previous section.
- Locations of each station (longitude and latitude) and the corresponding distances and travelling times between each station
- Cost of installing or removing docks including time, transport, manpower and manufacturing/disposal.

We also need to take into considerations the constraints of the problem as we are not being given an infinite amount of resources for this project. For example, it will be important to stay within the budget of the project, which is how much the company is willing to spend on the installation or removal of docks. Also, as discussed in the previous section, the presence of interconnected stations is a constraint; increasing the number of docks at once station will probably warrant increasing the number of docks for the other interconnected stations as well due to the constant traffic between them. One final constraint would be the limit on how many docks can be at one station. Docks and docked bikes take up space, but each station has a finite amount of space around it that will limit the number of docks that can be allocated to that station.

Overall, our optimization problem should help us find the the number of docks to be placed at each bike sharing station where the queue time is minimized, given data such as the location of each station, the time and distance between each stations, the cost of re-allocation of docks, and the outputs of our previous two models.

Given the long-term limitations of our initial ARIMA model, we should also re-run this optimization model at the same time we re-run the ARIMA model. This will enable us to monitor any changes in the solution since the last run.

This brings us to the end of the first part of our solution.

6. Experiment Design: Incentivizing Crowd Sourced Rebalancing

For the second part of the solution, we introduce crowd sourced rebalancing: in crowd sourced rebalancing, volunteers help to move a bike from a pick-up station to a nearby drop-off station and are rewarded. Generally, volunteers are prompted through a system (like a bike sharing app) to move a bike from a specific station to another specific station. If carried out properly, rebalancing through this strategy can be done at a very low cost to the company.

For a company trying to implement crowd sourced rebalancing for the first time, there would probably some experimentation to find the reward that will best motivate volunteers. Here are some factors that could have been considered:

- Type of rewards: What types of rewards are users motivated by? (eg. Cash balance, discounts on future trips, redeemable points)
- Pricing schemes: How much does the reward have to be before a volunteer is willing to move a bike? Should it be a fixed value per bike moved or dynamic pricing?
- Location of pick-up/drop-off stations: How far are volunteers willing to go to drop off a bike?

And data to be collected from each experiment:

- Number of bikes moved
- Pick-up/drop-off stations for each bike and distances between those stations
- User feedback, such as satisfaction rating for the reward scheme

We can combine different to create a number of solutions. These solutions can be rolled out on the bike sharing app, and the number and type of successful rebalancings recorded for each solution. Then we can conduct an experiment to arrive at the best solution.

Since there are multiple factors, A/B testing is out of the questions. We will have to pick an experiment design between a factorial design experiment and a multi-armed bandit algorithm.

A factorial design experiment allocates an equal sample size to each option to test all possible variations. It gathers data on all possible variations before presenting the final winner. With a multi-armed bandit algorithm, a solution is presented after every iteration, usually in real time. A multi-armed bandit algorithm is also designed to maximize gain/minimize loss during the experiment, while simultaneously collecting data to perform hypothesis testing.

Due to its ability to maximize the gains of the experiment, I believe the multi-armed bandit algorithm would be better than a factorial design experiment. Assigning equal samples to a lower performing solution like a factorial design experiment does means the company is missing out on potential volunteers that could be

helping to rebalance bikes, which means a potential loss of customers and business opportunities at a nearby station.

One of the downsides to a MAB is that it's a much more complex model to run than a factorial design experiment. Although it performs better, the company may not have the budget to make and maintain this solution.

7. Conclusion

In summary, here are the steps we took to increase the efficiency of bike sharing systems.

For optimizing the allocation of docks at stations:

1. Given bike trip records of start and endpoint data such as location and time, use an ARIMA model to predict demand at each station.
2. Given locations of stations, time and distance between each station, use the Louvain algorithm to identify interconnected stations that users frequently travel between.
3. Given the output of the earlier two models along with the available resources (company budget, fleet size), use an optimization program to optimize the allocation of docks with the goal of minimizing queue times for bikes within a budget constraint.

For incentivizing crowd sourced rebalancing:

1. Given potential solutions created by combining different factors, use a greedy multi-armed bandit experiment to pick to solution that will drive the most effective rebalancing.

Most of the data required for this project can be obtained from bike sharing app data and thus easily collected.

Regarding the re-running of models: we have already established that our optimization model should be re-run every six months to a year. However we may not always want to take action every time the solution changes - re-allocation of docks may be costly and if action is taken too often, it may cost more than the money it generates. Propose we set up a threshold, for example in the queue timings, so that when it goes over a certain threshold, we can be confident that re-allocation will be effective and the profit earned from it will far outstrip the cost of re-allocation.