Implementation of methods to predict demand, shortages, and evaluation of Resource allocation in the CitiBike System

Rohan Pitre(rp2816), Conrad de Peuter (cld2167)

1 Abstract / Introduction

Bike sharing systems are growing in popularity across the world. One main issue with bike sharing systems is station imbalance. This means that there are times when a rider goes to a station to pick up a bike and no stations are available, or that a rider goes to a station to return his bike and no docks are available. Data driven approaches are being used more and more to help the operations of bike sharing more efficient.

Currently, there is a lot of literature about predicting the demand of bike sharing systems and how to solve the optimization problems of rebalancing bikes at each station. However, these two methods haven't been applied simultaneously. Methods to solve the optimization problems assume a fixed demand rate which do not accurately represent how demand actually behaves. Furthermore, there are many heatmaps and dashboards depicting the current state of CitiBike, but none of these resources show predictions of future demand.

Our work will focus on integrating more realistic methods to predict demand in conjunction with quantitatively assessing strategies to rebalance the system. In particular, we will try to improve current methods to evaluate how balanced the system is by estimating out-of-stock events, as well as methods to assign routes to vehicles to move docks and bikes to have a more balanced system. Finally, we will explore approaches to encourage current riders to rebalance the system through promotions.

This problem is relevant to the course because we would like to build a system that can do this analysis in real-time. The current literature focuses on offline analysis and making these methods more efficient. Our goal is to integrate the current methods into one system that could potentially be used as a tool by bike-sharing systems to make decisions.

2 Data

2.1 Sources

2.2 CitiBike Data

2.2.1 Historical Trips

CitiBike provides trip-level data from 2013-2016.

2.2.2 Live Status

CitiBike provides an API to get real-time status at each station. Citibike also provides API access to its Bike Angels incentive programs' current status.

2.3 Weather Data

Weather data was collected using the Weather Underground API. Using hourly METAR measurements, we were able to get rainfall and temperature levels hourly from 2013 - 2016. This data was parsed and cleaned before integrating into models.

2.4 Cleaning

3 Models

3.1 Baseline

3.2 Station-based

The first model we attempted was a simple AR model, with a lag period of 1 week. We used the first 75% of each month to predict the last 25%. As you can see in the figures below, the high traffic stations were modeled fairly accurately by these models, but the low traffic stations had less-consistent trends and were harder to predict. Stations were generally much harder to model than neighborhoods as they have high variance. Also, this simple AR model doesn't take weather/holidays into account, which may have a large affect on station usage.The RMSLogE for the station models, using the first 75% of was .7569.

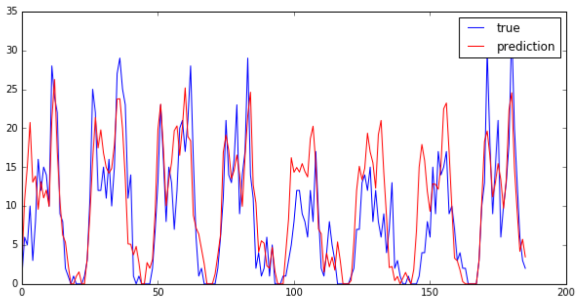

Figure 1: At the Great Jones St stationthe daily trends were consistent and easy to model

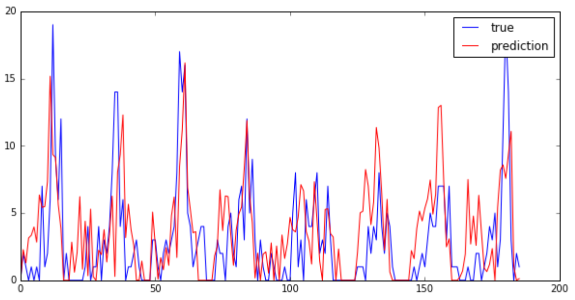

Figure 2: At the E 2 St & Avenue C station the trends were less consistent and not as easy to model

3.3 Neighborhood-based

Our baseline models for Neighborhood clustering were also AR models with a lag period of 1 week, trained in the same way as the simple station-based AR models. The RMSLogE for the cluster models was .9784.

3.4 Bike Angels Program

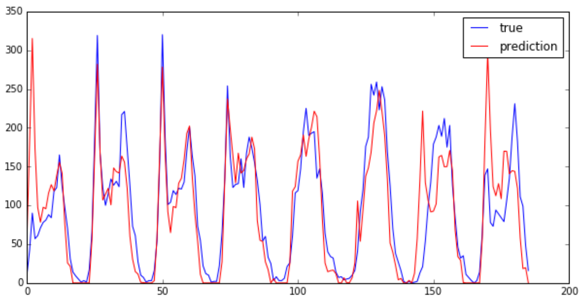

Figure 3: Modeling at a high use cluster

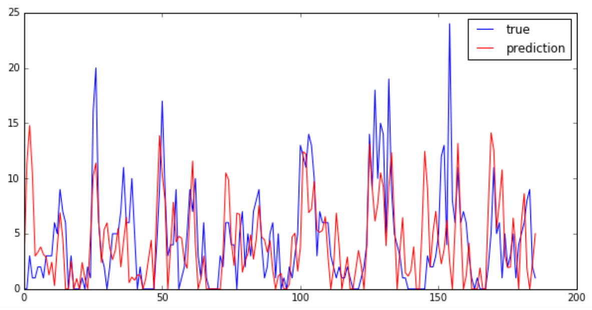

Figure 4: Lower traffic clusters were not as easy to model

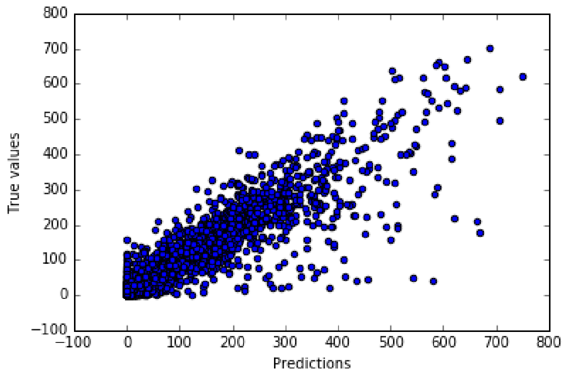

Figure 5: Simple AR for neighborhood-based models

4 Live Predictions on the Web

A live dashboard showing both the AR predictions we make and current system status is available to view at http://live.status.bike/. We include views for both Neighborhood level predictions and status as well as station-level. The predictions update every hour, and the current status updates as quickly as Citi Bike updates their live status. The web server was built using the Flask framework in Python. We forgo a SQL layer in favor of the Python Pandas for a lightweight way to merge the different data sources and format the output JSON in an easy-to-digest manner for the front end. The main function of the Flask app is to build API's for both the cluster and station-level predictions. The output of the /clusters api is GeoJSON, which integrates easily into the front end. The output of the /stations api is regular

JSON. Both APIs give statuses and predictions for the entities they describe, and are cached every 50 seconds to improve performance and scalability. The Flask app also returns the a small html file for the map, but all of the HTML rendering is done on the client. The front end was built using React in ES6 and transpiled to browser-readable Javascript using Babel and Webpack. The maps are build using the react-leaflet module. The whole web-stack (Node + Python) is created by a Docker image and is hosted on AWS Elastic Beanstalk, so deployment is very simple. In the current set up the predictions are using the current month of the previous year. Citi Bike publishes trip data quarterly. Because Citi Bike's live feed only includes system status, not current trips, we are not able to use the previous weeks' data to predict the current, and we felt using the current month of the previous year would give the best accuracy as opposed to data from up to 3 months before the current time. With this system models are rebuilt every month, and the new predictions are uploaded. If we had Citi Bike's live trip data to incorporate we would automate the process of re-building the models in a separate container, which would have the sole purpose of retraining models with an interval we determine, and uploading the data to the web so the prediction API's can display the new predictions seamlessly.

5 Impact

Live predictions can help with both Citi Bike and their restocking, as well as system users for trip planning.

6 Future Work

Going forward we would like to extend our analysis to other cities with similar bike share programs and open data. As long as a city has a similar current status feed and historical trip data, the analysis and website are easily extendible.

7 Related Literature

- *Background for the project:* Chen et al. introduce a Dynamic Cluster-based approach to predicting over-demand stations [2]. Li. et al. introduce a Gradient Boosting Regression tree model to predict the rate of demand going to and from an individual station. Zhang et al. introduce a regression model for individual trip destination and length predictions [1],. We plan on implementing these methods to find which are best at predicting demand in the system.

- *Work the project relies and builds on:* Some preliminary work has suggested a language for specifying query plans that we could borrow from. Also, Recent work nt ways to store and index data lend credence to the need for different cost models for access data in relations.

  Parikh et al. introduce a Markov Model to predict optimal restocking levels of a system [6]. Freund et al. consider the problem of optimal dock capacity [4].Raviv et al. also have an approach to determinig optimal inventory methods. We will use these approaches to evaluate how CitiBike restocks it's system.

  Schuijbroek et al. investigate optimal routing for system rebalancing in bike shares [5]. We plan on using their approach to evaluate the Citi Angels program.

8 Define Success

This project will be a success if we can accurately predict systems shortages ahead of time. After the models are fit, we will use them to predict real-time data. With better demand prediction we will be able to evaluate current resource allocation methods. If our model is able to correctly predict outages ahead of time, it will be a success. Finally, our web visualization will be useful for riders who expect to use the system, and the system operators to make strategic decisions by moving bikes and docks or offering incentives to riders .

References

[1] Zhang, J., Pan, X., Li, M., & Yu, P. S. (2016). Bicycle-Sharing System Analysis and Trip Prediction. 2016 17th IEEE International Conference on Mobile Data Management (MDM). doi:10.1109/mdm.2016.35

[2] Chen, L., Jakubowicz, J., Zhang, D., Wang, L., Yang, D., Ma, X., . . . Nguyen, T. (2016). Dynamic cluster-based over-demand prediction in bike sharing systems. Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing - UbiComp '16. doi:10.1145/2971648.2971652

[3] Li, Y., Zheng, Y., Zhang, H., & Chen, L. (2015). Traffic Prediction in a Bike-Sharing System. Proceedings of the 23rd ACM International Conference on Advances in Geographical Information Systems.

[4] Freund, D., Henderson, S. G., & Shmoys, D. B. (2016). Minimizing Multimodular Functions and Allocating Capacity in Bike-Sharing Systems. arXiv preprint arXiv:1611.09304.

[5] Schuijbroek, J., Hampshire, R., & van Hoeve, W. J. (2013). Inventory rebalancing and vehicle routing in bike sharing systems. Chicago

[6] Parikh, P., & Ukkusuri, S. V. (2014, August). Estimation of Optimal Inventory Levels at Stations of a Bicycle Sharing System. In Transportation Research Board 94th Annual Meeting (No. 15-5170).

[7] Raviv, T., & Kolka, O. (2013). Optimal inventory management of a bike-sharing station. IIE Transactions, 45(10), 1077-1093.