Passenger Saver based on Uber Surge Multiplier Final Report

PENG ZHENG*, Georgia Institute of Technology
JINLI YAN*, Georgia Institute of Technology
XIPENG CHAI*, Georgia Institute of Technology
HAOMIN LIN*, Georgia Institute of Technology
YANJUN DING*, Georgia Institute of Technology
YUNTIAN ZHANG*, Georgia Institute of Technology

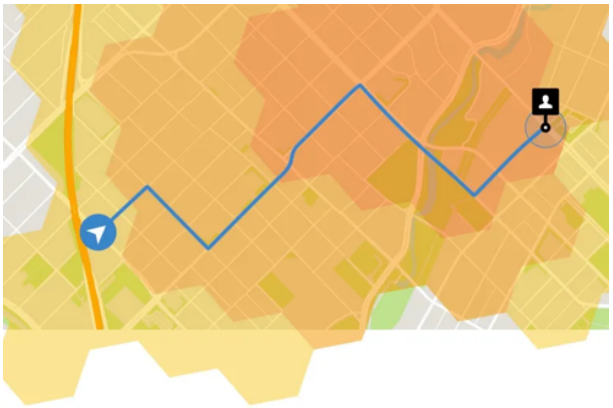


Fig. 1. Schematic Diagram of Distribution of Uber Surge Multiplier.(https://www.uber.com/us/en/drive/partner-app/how-surge-works/)

1 INTRODUCTION & PROBLEM DEFINITION

Uber is a technology company that is famous for managing a ride-sharing platform. One of the most important characteristics of Uber is the flexibility of supply[4].Today, what we will focus on is Uber's dynamic pricing called "surge multiplier".Moreover,Uber will charge passengers based on two factors: time and distance. During times of high demand, Uber will use this to

*All team members have contributed similar amount of effort.

Authors' addresses: Peng Zheng, peng_zheng@gatech.edu, Georgia Institute of Technology; Jinli Yan, jinli_yan@gatech.edu, Georgia Institute of Technology; Xipeng Chai, xchai30@gatech.edu, Georgia Institute of Technology; Haomin Lin, humaslin97@gatech.edu, Georgia Institute of Technology; Yanjun Ding, yding366@gatech.edu, Georgia Institute of Technology; Yuntian Zhang, yzhang3469@gatech.edu, Georgia Institute of Technology.

increase prices[3]. The surge multiplier will vary from region and time. **Fig. 1** quoted from the Uber website illustrates how this works. The darker the color is, the higher the price is. To be honest, this method may increase the efficiency of ride-sharing. With the help of surge, it reduces some customers out of the market, for example, many young students may give up calling a Uber when the price is relatively high. Besides, a higher price is able to attract more drivers to drive during the high-demand time or come to this region[3]. The surge multiplier will be updated every 5 minutes. It may sound reasonable, but this information is only visible for drivers. Passengers can be charged a higher price unconscious.

The problem is that surge multiplier information is inaccessible to passengers, and Uber/Lyft haven't revealed useful details about their Surge Algorithms. As a result, our purpose in this project is letting this information be accessible to passengers as well by estimating. For example,in **Fig. 1**, one person is on the edge of a high surge multiplier region. We will use data from the past to give him a guide to move to the nearest region with the lowest surge multiplier. To use more valid data, we decide to use the Uber data from San Francisco due to the largest number of Uber[5].

2 LITERATURE REVIEW

More and More researchers are interested in analyzing the spatial and temporal disparity of Uber demands[6, 7, 18], because Uber a representative of "sharing economy". Wang, et. al[18] studied the Uber accessibility by analyzing average wait time in different areas of Atlanta (**Fig. 2(a)**)[6, 7]. They tried to fit the data with models to figure out what factors (like wealth, race, public transportation etc.) influence the spatial disparity of Uber. Similar researches went on in New York City , which focused on analyzing number of pick-up data (**Fig. 2(b)**). Also, comparison between Uber and traditional taxi are made (**Fig. 3**), predicting that demands for Uber may exceeds that of taxi in the following years due to its convenience and relatively low price[6, 16]. Other reviewed papers are shown in the reference lists[9, 14].

Now that more and more users join the Uber ride, it's important for riders to know how the price of trip is calculated. As economist pointed out, truthfulness and information disclosure between riders and riding-sharing companies is vital to further development of ride sharing. Some researchers begin to pay attention on the surge mechanism of Uber[1, 3–5, 10, 15]. Chen, et. al tried to open a black box of surging price algorithms by applying a model to analyze Uber trip data based on cars supply, demand, wait times, locations, etc [3]. As they figured out, surge multiplier changes with areas and time (**Fig. 4**). It is refreshed every 5 minutes. Other researchers criticized the unfairness of surge, suggesting that surge price should be disclosed to public [3, 4].

Therefore, we want to make surge exposed to users while help them to choose a better place and occasion to book a Uber ride. In order to realize these functions, papers on map constructions are reviewed[2, 8, 11, 13, 17, 19] . Several studies are based on Google map API which is used to implement data visualization on maps [8, 13]. Brakatsoulas, et. al[2] studied

algorithms that exploit the trajectory nature of the tracking data which is helpful to us in designing a path leading to an area with lower surge multiplier.

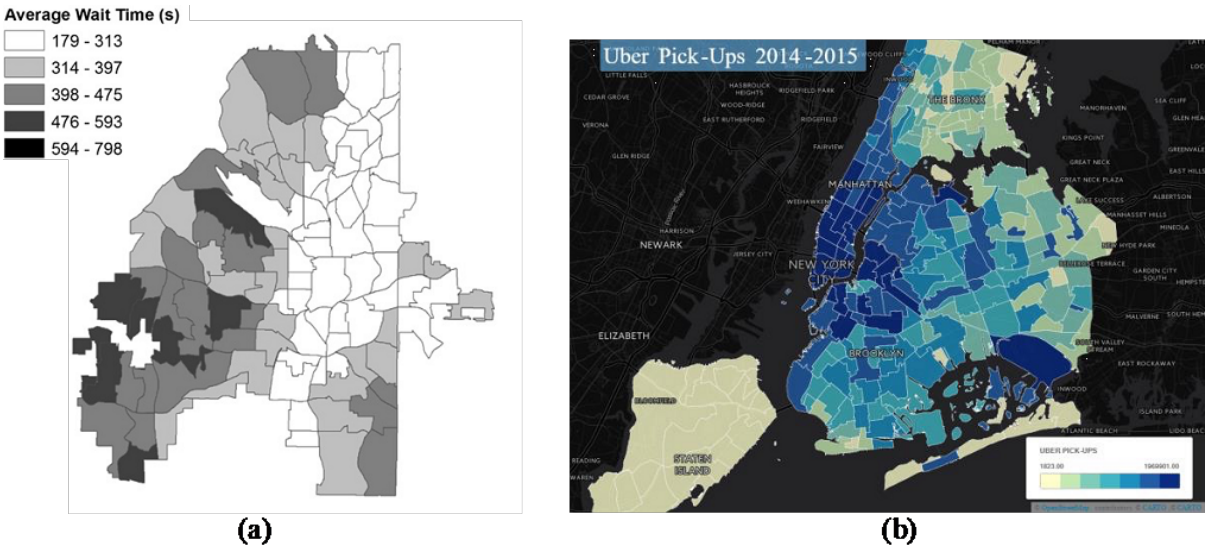


Fig. 2. (a) Estimated wait time of Uber in Atlanta in 2016 (reprinted from Wang, et. al, 2018) (b) Uber demands in New York city in 2014-2015 (reprinted from Correa, et. al, 2017)

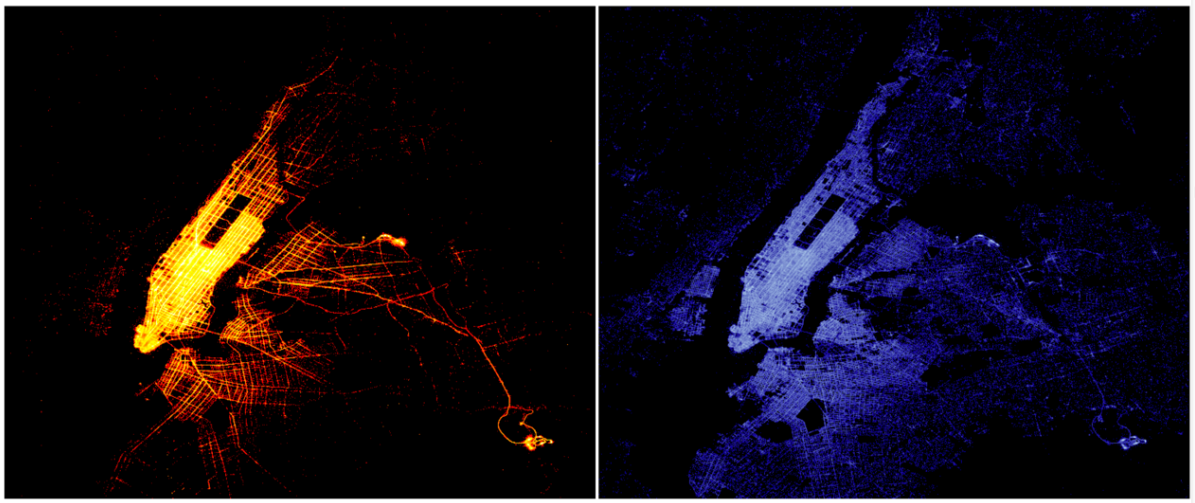
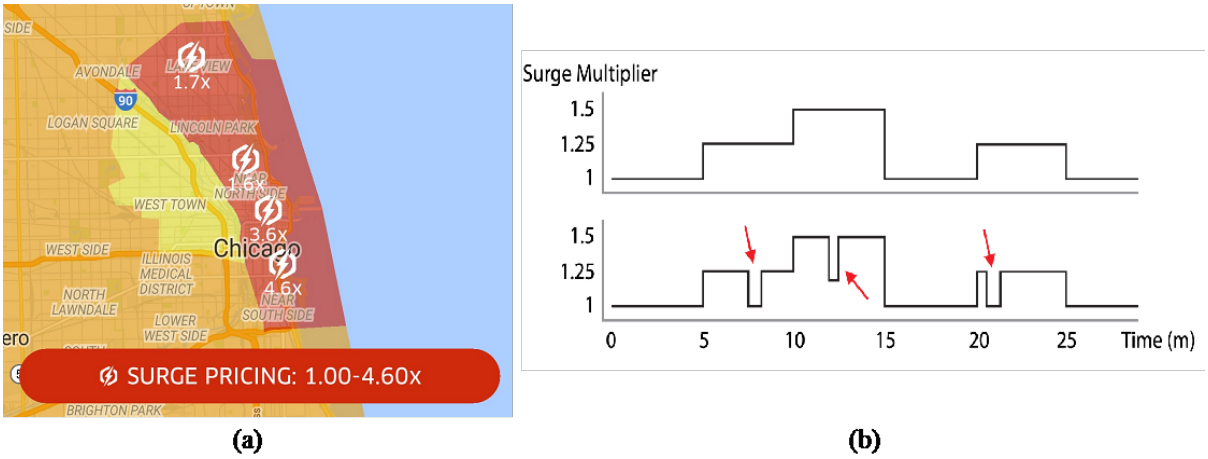


Fig. 3. Heatmaps of taxi and Uber pick-ups in New York City (reprinted from Correa, et. al, 2017)



(a) (b)

Fig. 4. (a) Surge multiplier changes with different areas (b) Surge multiplier changes with time (reprinted from Chen, et. al, 2015)

3 PROPOSED METHOD: INTUITION

At the beginning part of the project, we try to find something meaningful based on the current situation of Uber market. We found the surge concept, a concept related to price. Based on common sense, we know that this can be very sensitive to customers if they are provided with this kind of information.

However, after looking the current app and other related information, we discovered that the customers normally don't have access to this surge information. In contract, the admin and the drivers have access to these kind of information (In official Uber Driver app). These equalities put the customer in a disadvantageous position.

Under this circumstance, we burst out the initiative of doing this project: providing the customer with the surge information. That is the intuition of our project. For the current market, there are apps provide surge information, but what's new about our project is that we provide this information to the customers, which has never happened before.

By providing this surge information to the customers, there are some advantages based on these. First, customers can change (if possible) their order time based on the surge information we provide them. Therefore, our program can potentially help them save some money. Also, our program can also be helpful to reduce traffic pressure during rush hours.

4 APPROACHES

4.1 Data Cleaning

- **Data Set**:

  Due to close of Uber's public API, we have to use the data set "Uber and Lyft Dataset Boston, MA" from **Kaggle**. This data set contains 24 columns which can be divided into two parts. The first part of attributes is the basic information of a Uber or Lyft ride including timestamp, source, destination, cab type, product name, price, distance and etc. The other part mainly describes the weather when the trip was taken including temperature, visibility, short smmary of the whether and etc. The data set has some limitations that it just records the rides between some specific pairs of sources and destinations. Therefore, it's impossible for us to have a broad view of cab rides cases in whole Boston. Instead, we could only focus on rides between these hot spots.

- **Surge Multiplier Calculation**:

  In order to achieve our goal, predicting the surge parameter, we calculated the surge multiplier based on the price first. From the data we noticed that for a given source-destination pair, the price of the ride varies a lot. The product type matters because the launch price is different for various type of Uber or Lyft products (like UberX, Uber Black, etc). However, there're also some fluctuations of price for the same type of products. In this case, we expected that the price was multiplied by a surge parameter. Our surge multiplier is defined as:

$$s_i = \frac{p_i}{\sum_{j=1}^{n} p_j / n} \tag{1}$$

  where p_i is the price of ride i and the denominator is the average price categorized by source-destination pair and product type. With such classification, we can eliminate the impact of distance of ride or vary lauch price. By observation that the majority of data fluctuate around a specific value for a given group, we decided to use average price as our standard price(surge multiplier = 1). The final surge multiplier is rounded to an integer based on information given by Uber.
  Data cleaning work like eliminating NULL values and transfer timestamp into real time were run by Python *Pandas*. Preprocess works like calculating surge multiplier were performed on *Databricks* platform with *Spark/Scala*.

4.2 Data Analysis

- **Model & Algorithm**:

  A machine learning algorithm is helpful to us for predicting surge multiplier under different circumstances. In this project, we mainly use the random forests model. The key idea is to train a bunch of descision trees for bagging in order to reduce the variance. The algorithm is described as [12]:

Algorithm 1 Random Forests

For b = 1 to B:

Draw a bootstrap sample $\mathbb{Z}^*$

Grow a descision tree T_b to the bootstrapped data, by recursively repeating the steps below for each terminal node of the tree, until node size n_min is reached.

1. Select m variables at random from the p variables.
2. Pick the best variable / split-point among the m.
3. Split the node into two daughter nodes.

Output the ensemble of trees $\{T_b\}$

To make a prediction of a new set of attributes X:

Classification: Let $\hat{C}_b(X)$ be the class prediction of the bth descision tree.

Then $\hat{C}^B_{\text{rf}}(X) = majority\ vote\{\hat{C}_b(X)\}$

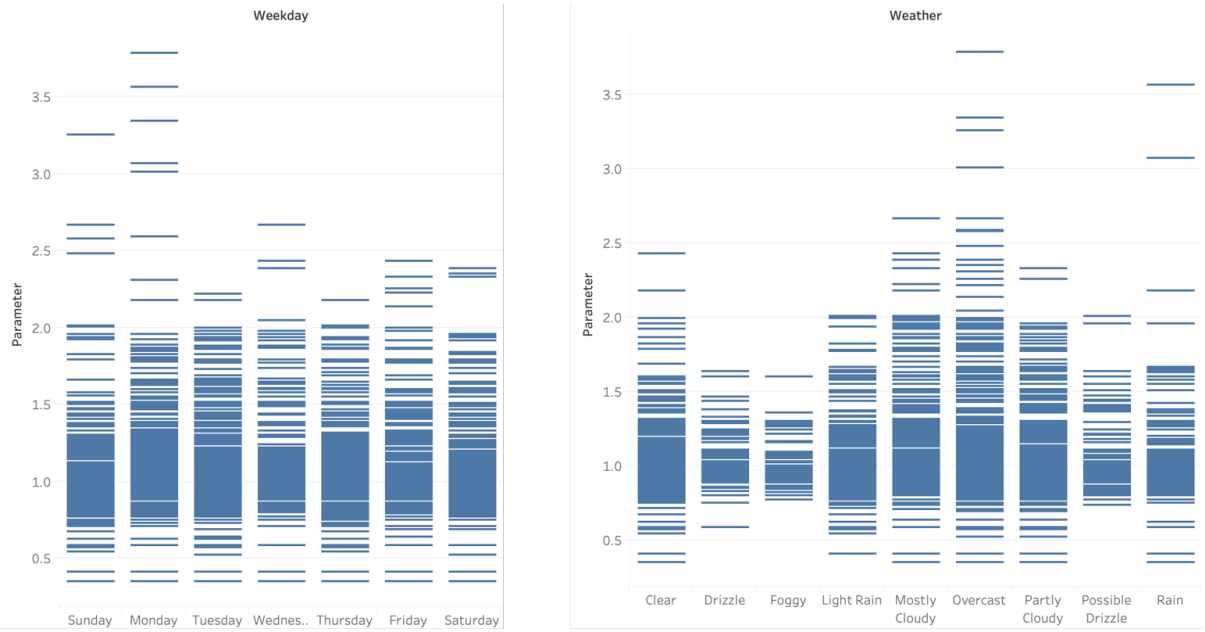
- **Training & Prediction**:

  In order to perform an efficient training and reduce the influence of noises and overfitting, 6 attributes were selected from total 24 columns which may have significent impacts on the surge multiplier. The attributes we focused on are { source, destination, distance, weather summary, weekday, hour }. It makes sense that surge multiplier increases on the rush hour of week days because of the huge demands for cab rides. Weathers like raining days may also increase the demands. Feature importance analysis were performed to see to what extent these variables influence our results. In **Fig. 5,Fig. 6** we made plots of surge multipliers and some attributes, showing their relationships.

  The accuracy our model is defined as:

$$accuracy = \frac{\sum_{i=1}^{N} \delta_{\hat{C}_i C_i}}{N} \tag{2}$$

$$\delta_{\hat{C}_i C_i} = \begin{cases} 1 & \text{if } \hat{C}_i = C_i \\ 0 & \text{otherwise} \end{cases} \tag{3}$$



(a) **(b)**

Fig. 5. Surge multipliers in (a) different weekdays (b) different weather.

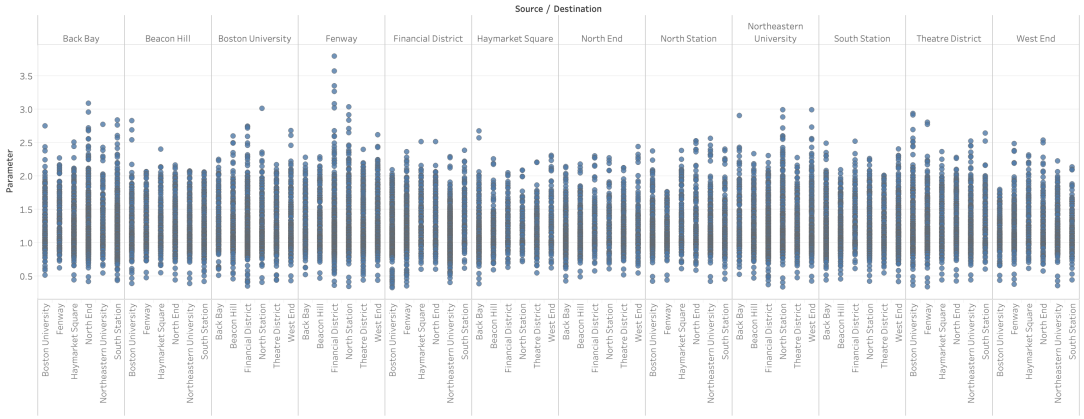


Fig. 6. Surge multipliers for different sources and destinations.

where N is the size of our test data set. Since it is a classification problem (the surge multiplier is an integer), we say the prediction is valid only if the prediction value equals to the original one.

The accuracy of the model is closely related to the parameters. We used gini index as our criterion of splitting and we focused on tuning number of descision trees and maximum depth of each tree. Details of our parameters and accurancy will be discussed in the following sections.

All machine learning stuff were performed with *Scikit-Learn*. To train the model with non-numerical attributes like strings, we mapped them to discrete numbers first with the help of *LabelEncoder* function. Then, *RandomForestClassifier* is chosen to fit our data set with 6 attributes selected. To tune parameters, we used *GridSearchCV* function to test different combination of parameters and search for the best accuracy scores.

With a proper model, we predicted the surge multiplier based on our self-constructed attributes set. The attributes set including all combinations of our selected attributes so that we could give our predictions under all circumstances.

4.3 User Interface

For concise performance of UI, we import PySimpleGUI in python to create a simple UI shown below (**Fig. 7**):

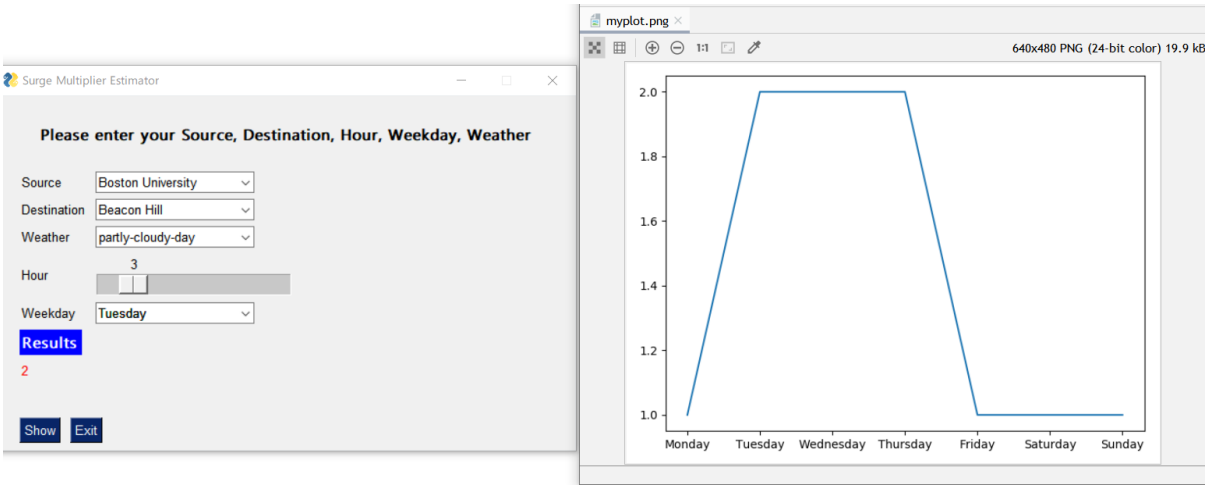


Fig. 7. Concise User Interface based on PySimpleGUI in python.

It has five factors for users to choose, Source, Destination, Weather, Hour, Weekday. After pressing Show Button, users can get a result of Estimated Surge Multiplier (ESM) (Normal(1), Medium(2), High(3)); moreover, it will pop up a diagram of ESM along with weekdays.

For embellished UI (**Fig. 8**), we use Adobe XD to design, as shown below, which is closer to an Application User Interface of iOS or Android; it includes user login, choosing factors, showing results & diagrams and returning.

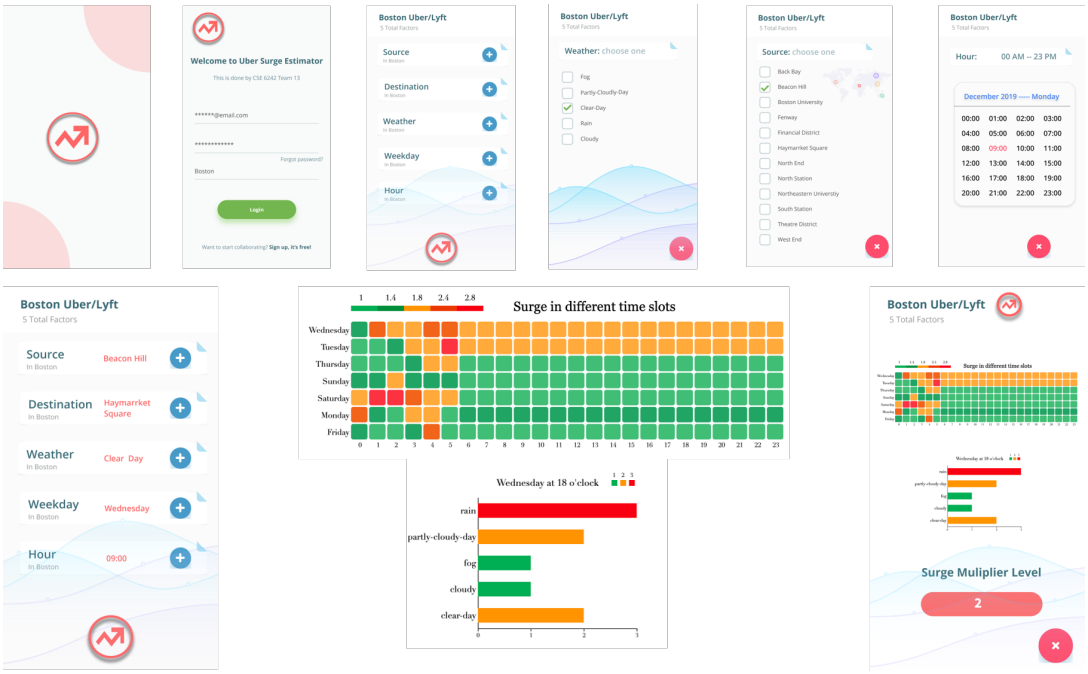


Fig. 8. Embellished User Interface based on Adobe XD UI designer.

5 EXPERIMENTS & EVALUATION

5.1 Testbed

There are several questions that the we want the experiment part can answer: (1) what kind of algorithm reach the highest accuracy? (2) In a specific algorithm, what kind of parameters can give the best prediction? (3) What is the final overall accuracy? (4) How each variable is related to the final surge information?

The tests for questions (1), (2) & (3) are simple and direct: By applying different algorithm and/or parameters and compare their final accuracy can we decide how the program can give the best possible outcome finally.

As for Question (4), we designed the test process after the program is trained. We validate the predicted surge information and the actual surge value and count its difference. Since there are different variables as input of the program (workday, source, destination, weather, etc.), we use control variate to avoid other variables. Also, the alternative way is to use the "feature-importance" during the training process.

In general, we divided the whole dataset (already cleaned one) into trained dataset (80%) and test dataset (20%). This division is for testing the accuracy in different dimensions.

5.2 Experiment Model

After data cleaning, we focus on establishing the model. We find that Random Forest model, Bayesian Regression model and *Support Vector Machine* are three basic and common model for machine learning. According to our survey, surge situation is not supposed to be linear with attributes, so Bayesian Regression is not a proper choice. Nor is SVM because it is efficient with datasets with relatively clear boundary. However, the surge parameter is a combination of different scenario. To verify our assumption, we try to run these three models on the dataset respectively and receive an accuracy of 48% for Bayesian Regression and 61% for Random Forest. We do not receive a result for SVM since there are too many attributes and the boundary is not clear. ***Based on the theoretical analysis and experimental performance, we choose Random Forest as our model.***

When building the model, we choose important attributes and use cross validation to optimize our model. First, we use all the 10 attributes to build a model and see the importance score. Despite source, destination and distance, we find weather, weekday and hour have higher importance rate than temperature and visibility. Therefore, we use 6 attributes to build the model: *source, destination, distance, weather, weekday and hour*. Then we use grid search to improve the model. We separate the dataset into 10 folds and apply cross validation method to find the model with best performance. We also tune the number of trees by training a random forest model with 300, 500 and 800 trees. We find that 500 has a better performance and takes reasonable time. Hence, we use 500 as the number of trees in the forest. ***With these chosen parameters, our model achieves an accuracy of approximately 75%.***

5.3 Experiment Visualization

After we finished tuning, we created a *csv* file that consists of all possible combinations of ***"destination", "source", "distance", "weekday", "hour" and "weather"*** that exist in the original database. This generated test set is used for predicting the surge multiplier for certain conditions chosen by the user of our system. Then we use pandas in python to import the test set and use s*cikit-learn* with our derived random forest algorithm to predict the surge multiplier of each combination. With the result set from our prediction, we aim to generalize all the situations for each specific hour in each specific weekday with different weathers. So we use *Spark* to calculate the average predicted surge multipliers for each case and set 3 ranges for the average multipliers to convert the final result to three distinct values *"1", "2", "3"*. Using the results, we create a new table that contains all the results.

This output can be used for visualization, so that we can see the general distribution of surge multipliers throughout all the combinations. And we can also observe the result of several specific conditions, which can help us judge the rationality of our prediction.

In the visualization part, we implement a ***heatmap*** with interactive bar charts in d3.js. In the heatmap, we show the average surge multiplier for all the weather conditions in that time slot, indicating how high the demand of cabs is in that time slot, which is shown here (**Fig. 9**) :

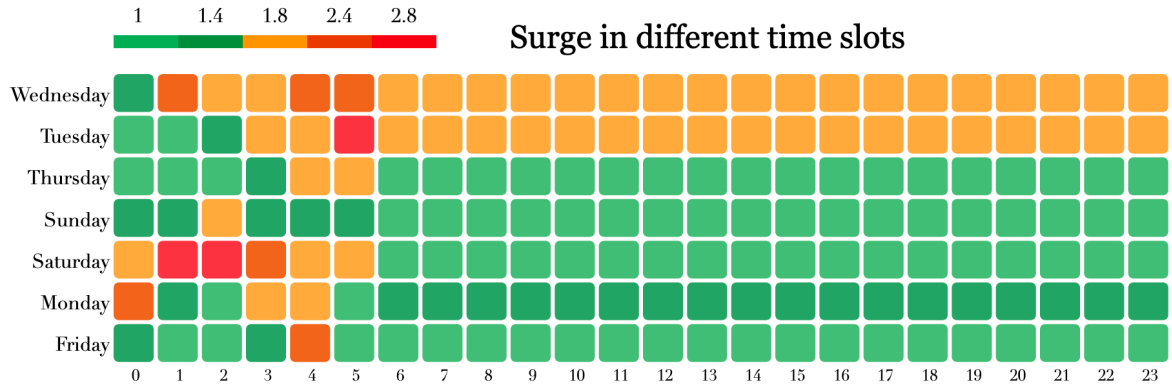


Fig. 9. Surge Multipliers change in different time slots.

And with mouseover function, we can show the specific surge multipliers for each weather in that timeslot, shown as (**Fig. 10**):

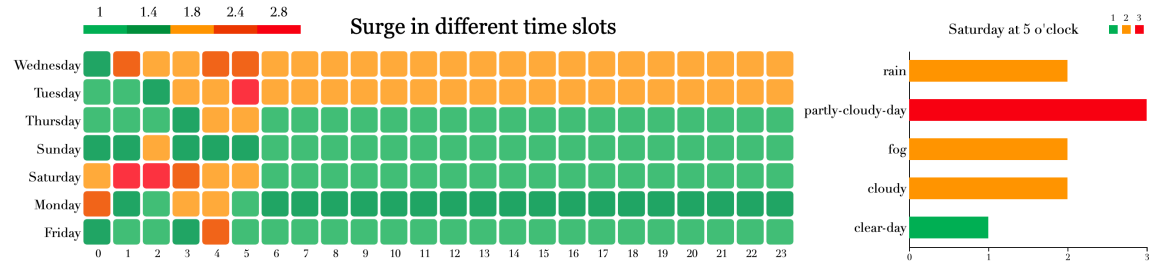


Fig. 10. Mouse-over shows a specific multipliers for each weather in a specific time slot.

With this visualization, we found several situations that can verify the rationality of our prediction.

For example, at 18 o'clock on Wednesday (**Fig. 12**), we can see the rainy day makes surge go up, for there can be more people in need of a ride. While fog days and cloudy days lead to lower surge multipliers, which can be the reason that those weathers make people less willing to go out and the surge multiplier go down.

Also, we can see that in the midnight, there can be many high surge multipliers in several timeslots. This phenomenon can be interpreted as the decrease of active drivers, so if there are considerable number of riders, it can push the multipliers up.

With several accountable cases, we can accept that our model is reasonable and produce reasonable results.

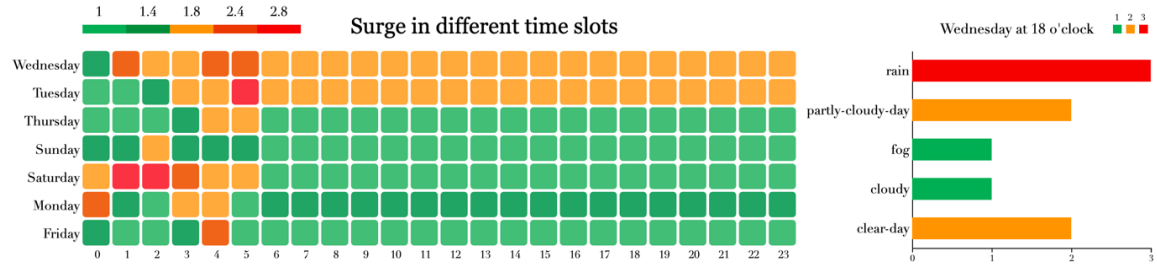


Fig. 11. Surge multipliers for rainy days at 18's o'clock.

6 CONCLUSION

Both Uber and Lyft have applied an strategy called surge multipliers in order to balance supply and demand during busy hours. However, what is concerned is that Surge Multipliers are inaccessible to the Public. As a result, our goal in this project is to make this accessible to passengers by estimating based on big data from Uber and Lyft. Due to complexities and impurities of datasets, we contributed a lot to data cleaning including reasonable surge-multiplier calculator, eliminating unreasonable data, etc. Then, we aim at training our datasets in order to achieve a high-accuracy prediction model. Based on the theoretical analysis and experimental performance, we choose Random Forest as our model to train on datasets of history rides. Finally, with chosen parameters (which have higher feature importance), our model achieves an accuracy of approximately 75% (80% for training, 20% for testing). What's more, we also commit to design interactive visualization (heatmaps) based D3.js; besides, an User Interface is also designed for the convenience of users. Conclusively, passengers can receive an estimated level of surges, interactive user interface as well as visualization based on our project.

ACKNOWLEDGMENTS

All team members have contributed similar amount of effort.

REFERENCES

[1] Niels Agatz, Alan Erera, Martin Savelsbergh, and Xing Wang. 2012. Optimization for dynamic ride-sharing: A review. *European Journal of Operational Research* 223, 2 (2012), 295 – 303. https://doi.org/10.1016/j.ejor.2012.05.028

[2] Sotiris Brakatsoulas, Dieter Pfoser, Randall Salas, and Carola Wenk. 2005. On Map-matching Vehicle Tracking Data. In *Proceedings of the 31st International Conference on Very Large Data Bases (VLDB '05)*. VLDB Endowment, 853–864. http://dl.acm.org/citation.cfm?id=1083592.1083691

[3] Le Chen, Alan Mislove, and Christo Wilson. 2015. Peeking Beneath the Hood of Uber. In *Proceedings of the 2015 Internet Measurement Conference (IMC '15)*. ACM, New York, NY, USA, 495–508. https://doi.org/10.1145/2815675.2815681

[4] M. Chen. 2016. Dynamic Pricing in a Labor Market: Surge Pricing and Flexible Work on the Uber Platform. 455–455. https://doi.org/10.1145/2940716.2940798

[5] Peter Cohen, Robert Hahn, Jonathan Hall, Steven Levitt, and Robert Metcalfe. 2016. *Using Big Data to Estimate Consumer Surplus: The Case of Uber.* Working Paper 22627. National Bureau of Economic Research. https://doi.org/10.3386/w22627

[6] Diego Correa Barahona, Kun Xie, and Kaan Ozbay. 2017. Exploring the Taxi and Uber Demand in New York City: An Empirical Analysis and Spatial Modeling.

[7] Sabiheh Sadat Faghih, Abolfazl Safikhani, Bahman Moghimi, and Camille Kamga. 2017. Predicting Short-Term Uber Demand Using Spatio-Temporal Modeling: A New York City Case Study. arXiv:stat.AP/1712.02001

[8] C. Fu, Y. Wang, Y. Xu, and Q. Li. 2010. The logistics network system based on the Google Maps API. In *2010 International Conference on Logistics Systems and Intelligent Management (ICLSIM)*, Vol. 3. 1486–1489. https://doi.org/10.1109/ICLSIM.2010.5461215

[9] Masabumi Furuhata, Maged Dessouky, Fernando Ordóñez, Marc-Etienne Brunet, Xiaoqing Wang, and Sven Koenig. 2013. Ridesharing: The state-of-the-art and future directions. *Transportation Research Part B: Methodological* 57 (2013), 28 – 46. https://doi.org/10.1016/j.trb.2013.08.012

[10] Nikhil Garg and Hamid Nazerzadeh. 2019. Driver Surge Pricing. arXiv:cs.GT/1905.07544

[11] Josh Greenfeld. 2002. Matching GPS Observations to Locations on a Digital Map. (01 2002).

[12] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. 2001. *The Elements of Statistical Learning.* Springer New York Inc., New York, NY, USA.

[13] Maged Kamel Boulos. 2005. Web GIS in practice III: creating a simple interactive map of England's Strategic Health Authorities using Google Maps API, Google Earth KML, and MSN Virtual Earth Map Control. *International journal of health geographics* 4 (10 2005), 22. https://doi.org/10.1186/1476-072X-4-22

[14] Farshad Kooti, Mihajlo Grbovic, Luca Maria Aiello, Nemanja Djuric, Vladan Radosavljevic, and Kristina Lerman. 2017. Analyzing Uber's Ride-sharing Economy. In *Proceedings of the 26th International Conference on World Wide Web Companion (WWW '17 Companion).* International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 574–582. https://doi.org/10.1145/3041021.3054194

[15] Kyungmin (Brad) Lee, Marcus A. Bellamy, Nitin Joglekar, Christo Wilson, and Shan Jiang. 2019. Surge Pricing on A Service Platform under Spatial Spillovers: Evidence from Uber. *Academy of Management Proceedings* 2019, 1 (2019), 16279. https://doi.org/10.5465/AMBPP.2019.16279abstract arXiv:https://doi.org/10.5465/AMBPP.2019.16279abstract

[16] L. K. Poulsen, D. Dekkers, N. Wagenaar, W. Snijders, B. Lewinsky, R. R. Mukkamala, and R. Vatrapu. 2016. Green Cabs vs. Uber in New York City. In *2016 IEEE International Congress on Big Data (BigData Congress).* 222–229. https://doi.org/10.1109/BigDataCongress.2016.35

[17] Nadine Schuessler and Kay Axhausen. 2008. Processing GPS Raw Data Without Additional Information. (01 2008).

[18] Mingshu Wang and Lan Mu. 2018. Spatial disparities of Uber accessibility: An exploratory analysis in Atlanta, USA. *Computers, Environment and Urban Systems* 67 (2018), 169 – 175. https://doi.org/10.1016/j.compenvurbsys.2017.09.003

[19] Xiaolu Zhou, Mingshu Wang, and Dongying Li. 2017. From stay to play – A travel planning tool based on crowdsourcing user-generated contents. *Applied Geography* 78 (2017), 1 – 11. https://doi.org/10.1016/j.apgeog.2016.10.002