Accurate Suggestion System For Transportation Provider

Menglin Liu (ml392@duke.edu)
Moyuan Li(ml390@duke.edu)
Vincentius Martin(vm76@duke.edu)

December 2, 2017

1 Introduction

Nowadays, the competition of traditional taxi is extremely struck by the sharing car, such as Uber and Lyft. This situation is caused by the usage of GPS and immediate real-time data analysis. However, traditional taxi is actually a safer choice for passengers. To increase the service of the traditional taxi, we would like to use the historical data of New York Taxi records to develop a tool for taxi vendors so that they could have a good idea about the demand of each area and lead drivers operate efficiently.
In our tool, the web APP will show a overview of the taxi demands given the pick up spots, time and weather. Our tool will generate estimations for the pick-up number for each zip-code area given the inputing information (time, location and weather). It will also mark the specific pick-up spots of the historical data given the fixed time in the Google map to guide vendors to modify their price.
Using the tool, taxi drivers could have a better guidance for where to go based on current condition, and thus increase profits. Especially, our system can provide taxi drivers with technical support and data analytics, which are already available to Uber and Lyft drivers with the companies' well-developed analytics platform. Our tool actually plays a role in filling the gap of the technical resources available for taxi drivers compared to Uber and Lyft drivers.

2 Related Work

[AMH+15] presents a system for effectively dealing with spatial data by doing partition across them. It adapts to the change in data and incrementally updating the partition.The data are being partitioned with k Nearest Neighbor. This approach will be useful for us as we will deal with large spatial data. Clustering the data will have a huge impact on how we can process data faster in order to give a timely recommendation to the user.

[ZCL+10] explains how they detect transportation modes from user-generated GPS logs. They use supervised learning to analyze these logs using features such as direction change rate, velocity change rate, and stop rate. The resulting deduction is able to give more than seventy percent accuracy to the transportation mode that GPS user uses. Our approach differs as we will use offline processing to detect not only users' preferred modes of transportation, but also their preferred locations during different time of the day.

[HWM+12] has similar objective with our project. It proposes a pick up tree based recommender system to minimize traveling distance for a given taxi before picking up its passengers. They apply clustering algorithm to GPS trajectory data and use the centroid as potential pick up point. They use gasoline consumption as weighted estimate to every route passed by the taxi. However, gasoline consumption is not an easy parameter to get. Our approach will instead utilize existing pickup and dropoff attributes from our database. These attributes consist of several parameters such as coordinate (e.g. longitude, latitude) and time.

3 Problem Definition

Taxi is one of the popular transportation options in New York city. However, with the development of Uber and Lyft, the competing power of traditional taxi becomes undermined. This may be caused

by the following reasons:

- Fixed fare of taxi
- Lack of knowledge of customers behavior and preference

Thus, we want to design a tool for taxi drivers and taxi vendors to lead drivers to the places where they could possibly make more profits. Afterwards, we will conduct further research to help them adjust the price according to the changing demands in the future.

3.1 Suggested pick up area for driver

We will offer a overview of the historical data to the driver, and it will include the popular pick up spots, the average number of requests, the most popular destinations, average distance for a given place and time. With this overview, driver may have a good sense of the distribution of demands. Then we will build a model based the historical data to predict the pick-up number for each zip-code area given time, temperature and precipitation. With this model, we will offer a searching tool, where drivers can input weekday, time, temperature, precipitation and their preference for pick-up area. Finally, the tool will offer an estimation of pick-up number for each specified area for the driver. The equation will expressed as following:

$$Numberof Pickups = f(weekday_d, hour_h, destination_{i,j}, temp_t, precipitation_p)$$

Where d represents weekday from Monday to Sunday; h presents the time from 0 o'clock to 23 o'clock; i, j represents the latitude and longitude of the place, t represents temperature in F, and p represents the precipitation in inch.

3.2 Comparison of Rush Hour and Non-Rush Hour

In order to analyze the difference in rush hour and non-rush hours, we decide to first visualize the pickup occurred during the morning rush hours (7-10 am) versus non-rush hours. We used the data for yellow cabs and green cabs, and read the monthly data sets into Spark R. Using only the spark engine spatially aggregate the pickup and drop-off locations and count the number of pickups or drop-offs that occurred in that location. We produced a plot of all yellow and green cab pickups together (with points colored by cab type) for rush hour and non-rush hour. Comparing the two plots, we can examine New Yorker commuting patterns with regard to using taxis, which could help taxi drivers to learn more about the consumer behavior. From Figure 1 we can observe that in non-rush hour, taxi pickups tend to cover a larger area around NYC, compared with rush hour. From the visualization, it appears that rush hour might be significant in predicting the number of pickups, therefore we incorporated rush hour as a feature in the model.

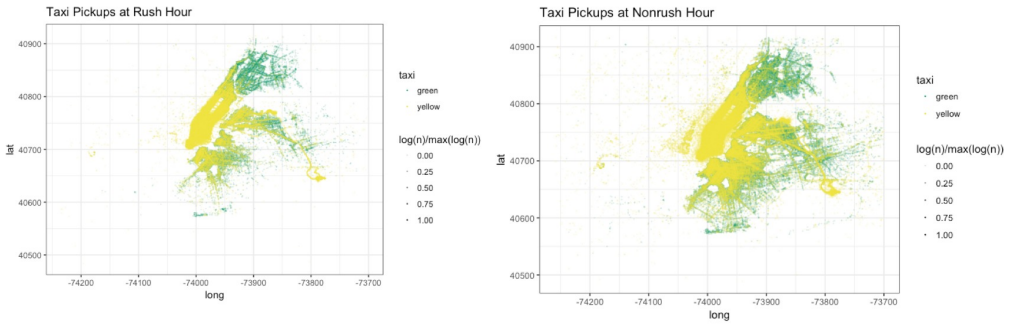

Figure 1: Taxi Pickups in Rush hour vs Non-rush hour

4 Algorithms

In order to achieve better prediction performance, three prediction models are explored, including linear regression, random forest, XGBoost. Linear regression allows us to exploit linear patterns in the data set. This model is an appealing first choice because variables are easy to interpret. Its also computationally efficient for large datasets. Random forest is robust against outliers and can prevent overfitting. XGBoost is an algorithm based on gradient boosting. It is a highly sophisticated algorithm, and its robust against all kinds of irregularities of data.

4.1 Prediction Taxi Demand Using Linear Regression

The following is the linear regression algorithm: Given paired data $(x_1, y_1), \cdots, (x_n, y_n)$, we model $Y_j = \beta_0 + \beta_1 x_1 + ... + \beta_p x_p + \epsilon$ where ϵ is random noise with $\mathbb{E}(\epsilon) = 0$ and $\mathbf{Var}(\epsilon) = \sigma^2$. we can get βs by minimizing the sum of the squared errors:

$$\sum_{j=1}^{n}(y_j - \beta_0 - \beta_1 x_1 .. - \beta_p x_p)^2.$$

4.2 Prediction Taxi Demand Using Random Forest

The following is the random forest algorithm:

- 1.Grow a forest of many trees.
- 2.Grow each tree on an independent bootstrap sample from the training data.
- At each node: (1) Select m variables at random out of all M possible variables (independently for each node).

  (2). Find the best split on the selected m variables.
- 4.Grow the trees to maximum depth (classification).
- 5.Vote/average the trees to get predictions for new data.

4.3 Prediction Taxi Demand Using XGBoost

Figure 2 is the XGBoost algorithm:

Algorithm: Cost-sensitive XGBoost

Require: Training set D=$\{(x_i, y_i)\}_{i=1}^{n}$, a differentiable cost-sensitive loss function $\Psi(y, F(x))$, *regularization term* Ω, δ, η, number of iteration M

1: *Initialize base learner* F_0;
2: **for** $t = 1$ *to* M **do**
3: *Calculate the gradient on* $\Psi : g_t(x_i) = 2\delta(y_i - p(x_i))$;
4: *Calculate the Hessian on* $\Psi : h_t(x_i) = -4\delta^2 p(x_i)(1 - p(x_i))$;
5: *Determine the tree structure* $\{q_{jt}\}_{j=1}^{T}$ *by maximizing* :

$$Gain = \frac{1}{2} \times \left[\frac{\left(\sum_{i \in I_L} g_i\right)^2}{\sum_{i \in I_L} h_i + \lambda} + \frac{\left(\sum_{i \in I_R} g_i\right)^2}{\sum_{i \in I_R} h_i + \lambda} - \frac{\left(\sum_{i \in I} g_i\right)^2}{\sum_{i \in I} h_i + \lambda} \right] - \gamma \ (see\ Eq.(24));$$

6: *Determine the optimal leaf weight* $\{\omega_{jm}\}_{j=1}^{T}$ *given* $\{q_{jt}\}_{j=1}^{T}$ *by*

$$\omega_{jt}^{*} = argmin_{\omega_j} \left(\sum_{i \in I_{jt}} \Psi(y_i, F_{t-1}(x_i) + \omega_j) + \Omega(\omega_j) \right);$$

7: $f_t = \sum_{j=1}^{T} \omega_{jt}^{*} I(x_i \in q_{jt})$;
8: $F_t(x) = F_{t-1}(x) + f_t(x)$;
9: **end for**
10: **return** $F_M(x)$

Figure 2: XGBoost Algorithm

5 Implementation

5.1 Data Processing and Feature Engineering

The original dataset consists of information about the yellow and green taxi and Uber pickups in May of 2014. The variables includes longitude, latitude, time of the pickup and the cab type etc. The original data has 23080677 rows, which is almost impossible to compute considering the limited computational resources. Therefore, we sampled the original datasets to reduce the dimension to 7000 rows. Then, we filtered the data with latitude between the 40 and 41.5, and longitude between -75 and -72. This makes sure that the dataset only includes taxi pickups inside central New York City area.

Hourly weather information is incorporated in the taxi dataset, as taxi demand are usually associated with weather. In order to deal with the missing data issues, we deleted the observations with missing values in temperature and rain. We then used the Python package geopy to look up the zip code based on the latitude and longitude of the pickup location. Then, the dataset is filtered to only include the observations with zip code in NYC, by comparing the computed zip code with the actual set of zip codes in NYC. This would prevent the possible mistake of including zip codes that are actually not in NYC.

As we think the weekday and hour in the day would have an impact on the number of pickups, the date feature in the original dataset is parsed into additional features like weekday, hour and rush (rush hour). For modeling purpose, we treat each of the 98 zip codes in the dataset as a dummy variable, which have value 1 if the pickup is at the particular zip code. For the purpose of cross validation and model evaluation, the data are divided into training set, which has 80% of the data, and testing set, which has 20% of the data.

The features that are included in the dataset for model fitting are as follows:

- Weekday, which has integer values in [0,6]
- Hour, which has integer values in [0,23]
- Rush, which is a Boolean variable. It has value True if the hour is between 7am to 10am. Otherwise, the value is False
- Temp(temperature), float variable measured in Fahrenheit.
- Rain (hourly rain precipitation), float variable, measured in inch.
- 98 zip code dummy variables, which only have value 1 if the pickup is at the particular zip code. Otherwise, the value is 0.

5.2 Model Fitting

Linear regression is fitted using "linear model" from "sklearn" package in python. Random forest is fitted using RandomForestRegressor from sklearn package. The parameters are tuned using cross validation to minimize the mean squared error. XGBoost is is fitted using the xgboost package. The hyper-parameters are tuned using cross validation to minimize the mean squared error.

5.3 Web UI

We finally built a web application tool to present our analysis results. To improve the performance of the web APP(i.e. output the results as soon as possible), we pre-proccessed the historical data in case that the user need to wait the result for several days if we use most updated data. In our application, after user input key fields to search such as zip code, temperature etc., the UI back-end will use the user input to call python predicting script. This script will then use the processed data to run predicting model. Finally the results will be sent back to the UI side waiting to be called, and be presented to the user. The complete process is shown as in the figure 3. As for the user interface, users (taxi driver) can input their current information (such as weekdays(0-6), time(0-23), rain(precipitation inch), temperature(F) and where want to go(zip-code)) to search for estimated number of pickups for taxi. A screen shot of the UI web is in the figure 4.

As for the results, we offer two modes for input: In the first mode, the required input are just weekday and time, then APP will return a table of estimated demand for each zip-code in New

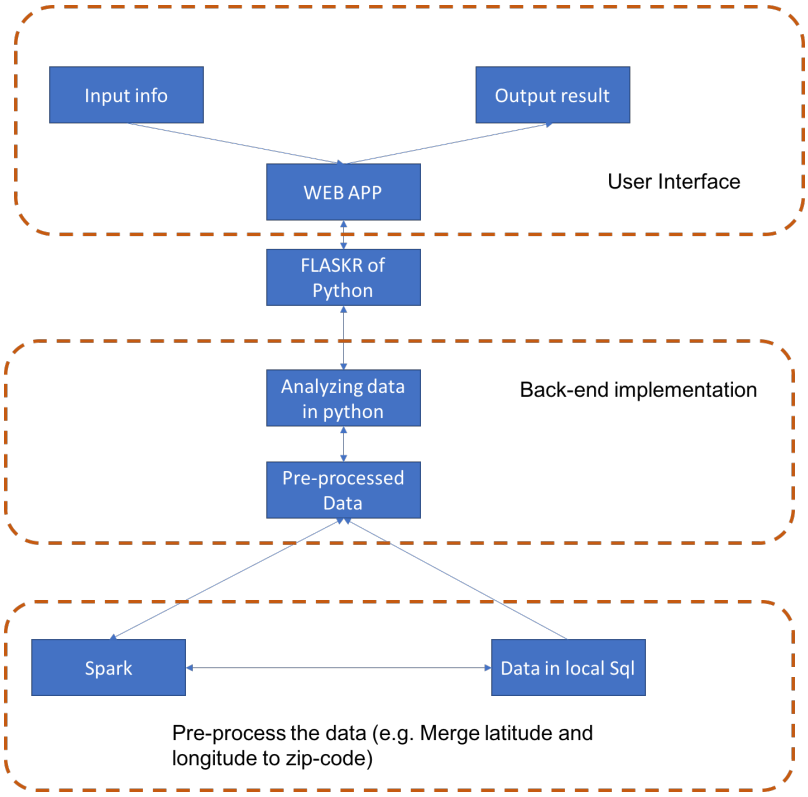

Figure 3: Architecture of implementation

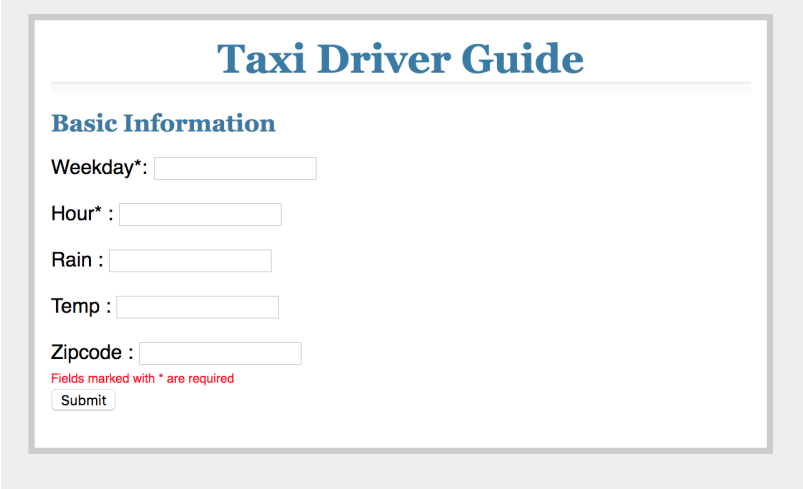

Figure 4: Inputing Screen

York. The result is not as specific as the second mode, In the second mode, when all fields are filled, the APP will return the estimated demand for the specified zip-code based on the weather condition such as temperature and rain condition that user offers. In this case, the user will directly see the accurate estimation for the place where he/she wants to go. Besides, the APP will also visualize the summarized historical pick-up data in a Google map. It will mark each transaction's pick-up spot with green dot in the Google map screen. User can drag and zoom in the Google map, to explore the specific pick-up spot. The pick-up locations shown in the Google map are filtered by user's inputing (weekday and time). The APP will only show the pick-up locations in the historical transactions happened in specified weekday and hour. The overview of UI screen is shown as in the Figure 5.

The front-end side was written in HTML formatted by CSS. The back-end side used Python to call the prediction function which were written in Python collaborated with spark R. The communication between front-end and back-end depends on "flaskr" package in Python.

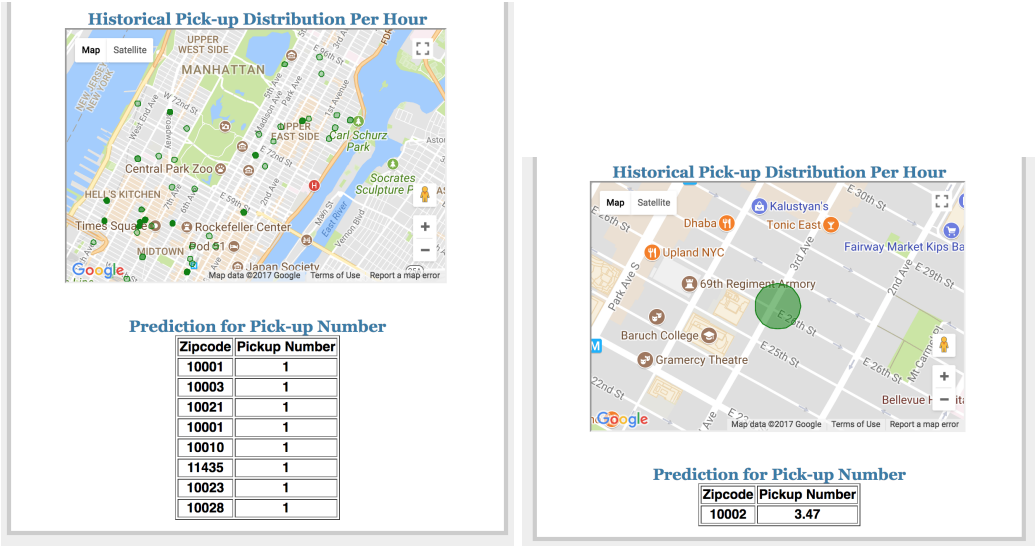

Figure 5: Results Screen

6 Experiment and Observations

6.1 Overview of the Dataset

Since the original NYC taxi dataset is huge, we just download and use the related part from it. After cleaning, there are total 23,080,677 tuples in our dataset. Here are the variables that are stored in the dataset.

- Taxi pick-up date and time
- Taxi drop-off date and time
- TLC taxi zone of the pick up location
- TLC taxi zone of the drop off location
- Elapsed trip distance in miles
- Time and distance fare calculated by the taxi meter

We also look into the distribution of green and yellow taxi pickups in Figure 6. This is done to further corroborate the feasibility of our prediction.

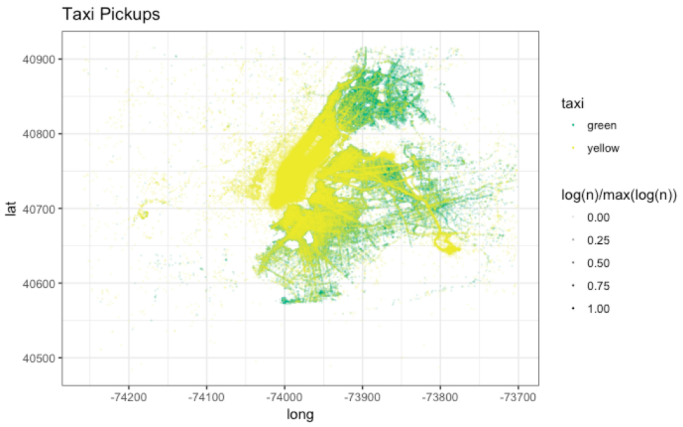

Figure 6: Taxi pickups

6.2 Model Evaluation and Performance Comparison

The three models are evaluated by cross validation. Mean squared error(MSE) is calculated for both training dataset and the testing dataset.

Here is a table of the comparison of train and test MSE for three models.

model	**train MSE**	**test MSE**
Linear Regression	1.769	2.634
Random Forest	0.126	1.32
XGBoost	1.364	1.154

Table 1: Model Evaluation and Comparison

It seems that both random forest and XGBoost have good performance in prediction. For linear regression, although the model is simple and easily interpretable, the predictive performance is the worse among the three models. The test MSE 2.634 is relatively large. Among the three models, the random forest model have the lowest train MSE, and fairly good test MSE. For the purpose of web user interface, we chose the random forest model to perform the prediction.

6.3 Qualitative Evaluation - Feature Importance

From the output of the random forest model, we can get the relative feature importance, which could provide some insights about which features are more important in predicting the number of pickups. From the Figure 7, it can be observed that Rain is the most important feature, compared with other features. This actually makes sense, as the taxi demand tend to increase when it rains. What surprises use is that it seems that rush doesnt provide too much information for predicting the number of pickups. The reason why "rush" is not significant predictor might be that the Hour feature is already in the model, which has similar information as in "rush". Therefore, when "Hour" already accounted for the effect of time in the model, "rush seems to be insignificant. In addition, temperature and weekday are also very important variables for prediction. From the prediction model, the temperature has a negative effect in the number of pickups. Compared with common sense, it's justifiable that people tend to take cabs when the weather is cold. There are also some certain areas which have more pickups than the others, such as the areas with zip code 10018, 10001,11040, and 10036, and 10014. By looking up these zip codes, these areas are around downtown Manhattan.

6.4 Comparison with other previous approaches

Compared with the previous work, our system will offer a direct suggestion and market overview for taxi driver at specific time and location. Besides, we will also have a better visualization for the final results as a picture, which is more efficient to help taxi drivers to make decisions where

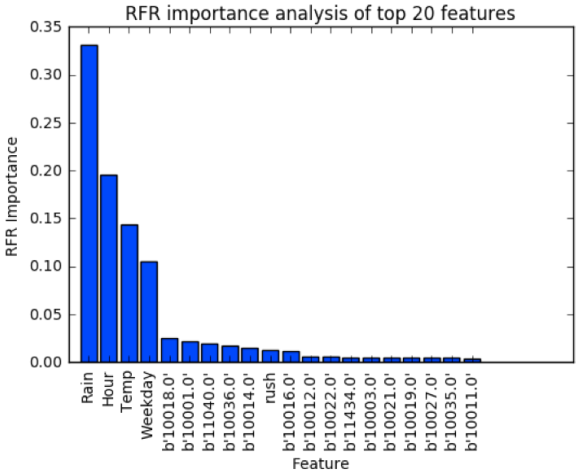

Figure 7: Feature Importance

to go and how much should charge.

7 Future Work and Conclusions

7.1 Suggested adjustment fare

Our work can be extend to suggest adjustment to the taxi fares. The demands for taxi are varying across different time and place. Usually the demands will increase during rush-hour. Correspondingly, the fare of Uber and Lyft will increase if the demand is much higher than the supply at specific area. However, the fare of taxi is always fixed, thus taxi drivers have no motivation to drive in those heavy-traffic roads. We plan to offer suggested fare adjustment at given area and time via analyzing the database of NYU taxi historical records. On the one hand, this strategy will increase the profitability of taxi companies and drivers, on the other hand, this strategy will attract drivers to those popular spots thus meet the increased demands. The equation of this problem would be expressed as:

$$Profit_{ijt} = f(fare\_amount|ij, demand_{ijt}, time_t, congestion_{ij}, averagedistant_{ijt})$$

Where i, j represents the latitude and longitude of the place, t represents time, congestion indicates whether area (i, j) is congested, and average distance represents how far the passengers want to travel at place (i,j) and time t.

7.2 Conclusion

In this project, we explored different predictive models to predict the taxi demand in NYC, and provide suggestions for the taxi drivers to possible pickup spots to maximize their profitability. The Web APP is not only supported by well-tested predictive models, but also provides a good visualization of the predicted pickup spots, which makes it convenient for drivers to use.

Member	Job description
Menglin Liu	System implementation as a web application
Moyuan Li	Data processing, model fitting and model evaluation
Vincentius Martin	Data analysis to get pick up trend and price adjustment. Getting more study from previous works.

Table 2: Job division for each team member

8 Work Division

9 Reference

References

[AMH+15] Ahmed M. Aly, Ahmed R. Mahmood, Mohamed S. Hassan, Walid G. Aref, Mourad Ouzzani, Hazem Elmeleegy, and Thamir Qadah. Aqwa: Adaptive query workload aware partitioning of big spatial data. *Proc. VLDB Endow.*, 8(13):2062–2073, September 2015.

[HWM+12] Haoran Hu, Zhiang Wu, Bo Mao, Yi Zhuang, Jie Cao, and Jingui Pan. *Pick-up tree based route recommendation from taxi trajectories*, pages 471–483. Springer International Publishing, 2012.

[ZCL+10] Yu Zheng, Yukun Chen, Quannan Li, Xing Xie, and Wei-Ying Ma. Understanding transportation modes based on gps data for web applications. *ACM Trans. Web*, 4(1):1:1–1:36, January 2010.