

Mega Taxi

Lingyun Gao, Hongyue Lan, Yingxin Zhang, Tangyao Zhao, Sharon Cui

A predictive technology startup

Team Members



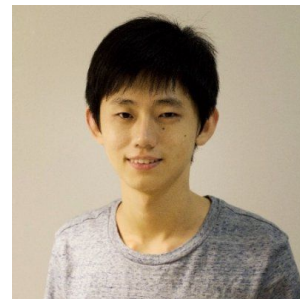
Sharon Cui



Tangyao Zhao



Lingyun Gao



Hongyue Lan



Yingxin Zhang

Overall Architecture

We are a new taxi Startup aiming to use **predictive technology** to solve NYC's taxi **pick up** and **pricing issues**.

Long waiting time and **no available pickup taxis** are two major issues for Manhattan taxis.

With the **unmaching of supply and demand** of the taxi, the pricing issue is important for us to tackle.



Overall Architecture

Problem: The accurate prediction of **popularity of pick up location** for the traditional taxi in NYC-Yellow Cab is missing.

The pricing of taxi is not reasonable without consideration of popularity of the **picking up needs** such as popular events, rush hours etc., or the environmental condition such as weather.

Solution: Use **KRR** to predict picking up times in different locations in Manhattan for Yellow Cab according to the relationship between the **historical pickup time** and the **weather** in Manhattan.

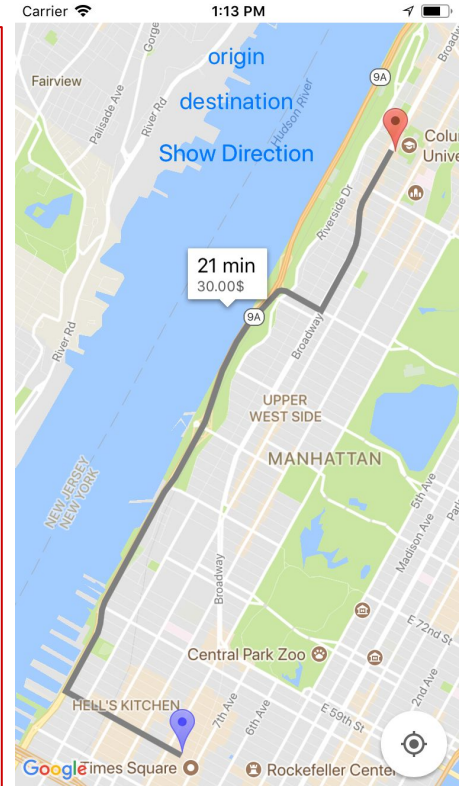


Overall Architecture

Created a Taxi calling app:

Embedding Google Map SDK for iOS, we input the trip duration and distance.

Using the same model with cost parameters, can predict the cost of the trip



Data

- **NYC Yellow Taxi Data:**
 - Yellow Taxi csv files: 2015.07 - 2016.06
 - http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml
- **Past Weather Data:**
 - <https://www.timeanddate.com/weather/@8479493/historic>

Data

Raw Data for NYC Yellow Taxi (format for each csv file is the same):

	VendorID	tpep_pickup_datetime	tpep_dropoff_datetime	passenger_count	trip_distance	pickup_longitude	pickup_latitude
0	2	2015-01-15 19:05:39	2015-01-15 19:23:42	1	1.59	-73.993896	40.750111
1	1	2015-01-10 20:33:38	2015-01-10 20:53:28	1	3.30	-74.001648	40.724243
2	1	2015-01-10 20:33:38	2015-01-10 20:43:41	1	1.80	-73.963341	40.802788
3	1	2015-01-10 20:33:39	2015-01-10 20:35:31	1	0.50	-74.009087	40.713818
4	1	2015-01-10 20:33:39	2015-01-10 20:52:58	1	3.00	-73.971176	40.762428

dropoff_latitude	payment_type	fare_amount	extra	mta_tax	tip_amount	tolls_amount	improvement_surcharge	total_amount
40.750618	1	12.0	1.0	0.5	3.25	0.0	0.3	17.05
40.759109	1	14.5	0.5	0.5	2.00	0.0	0.3	17.80
40.824413	2	9.5	0.5	0.5	0.00	0.0	0.3	10.80
40.719986	2	3.5	0.5	0.5	0.00	0.0	0.3	4.80
40.742653	2	15.0	0.5	0.5	0.00	0.0	0.3	16.30

Data

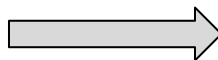
Data Cleansing (Taxi Data):

- **Keep the following features:**
 - Pickup/dropoff time,
 - pick up location
 - trip duration
 - total fare amount
 - passenger count

Data

- Remove dirty records (e.g. nulls, negative price, payment type except card and cash, etc).
- Keep longitude and latitude(rounding to the nearest thousandth)

pickup_longitude	pickup_latitude
-73.993896	40.750111
-74.001648	40.724243
-73.963341	40.802788
-74.009087	40.713818
-73.971176	40.762428

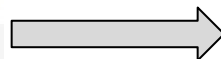


pickup_longitude	pickup_latitude
-73.994	40.750
-74.002	40.724
-73.963	40.803
-74.009	40.714
-73.971	40.762

Data

- Keep `tpep_pickup_datetime`, remove `tpep_dropoff_datetime`
- Add a new feature “duration”, unit: seconds

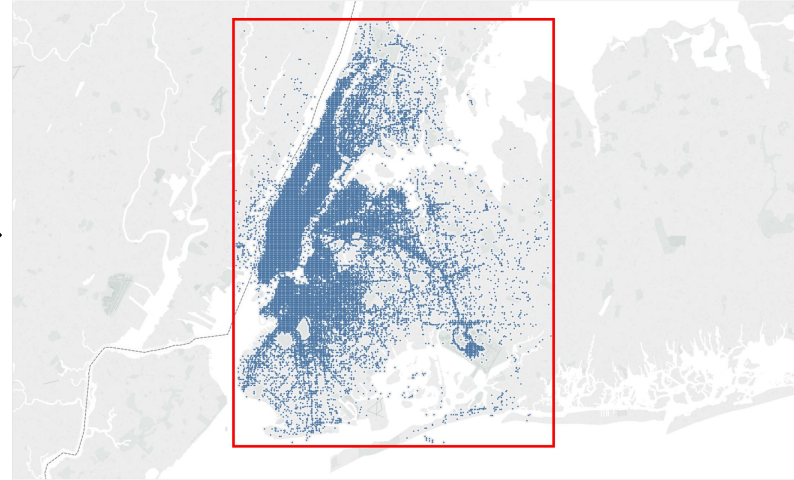
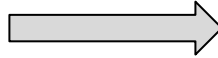
<code>tpep_pickup_datetime</code>	<code>tpep_dropoff_datetime</code>
2015-01-15 19:05:39	2015-01-15 19:23:42
2015-01-10 20:33:38	2015-01-10 20:53:28
2015-01-10 20:33:38	2015-01-10 20:43:41
2015-01-10 20:33:39	2015-01-10 20:35:31
2015-01-10 20:33:39	2015-01-10 20:52:58



<code>tpep_pickup_datetime</code>	<code>duration</code>
2015-01-15 19:05:39	1083.0
2015-01-10 20:33:38	1190.0
2015-01-10 20:33:38	603.0
2015-01-10 20:33:39	112.0
2015-01-10 20:33:39	1159.0

Data

- Eliminate wrong pickup and dropoff locations by setting a rectangular boundary



Data




- Result (reduced half of the size):

	tpep_pickup_datetime	passenger_count	trip_distance	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude
0	2015-01-15 19:05:39	1	1.59	-73.994	40.750	-73.975	40.751
1	2015-01-10 20:33:38	1	3.30	-74.002	40.724	-73.994	40.759
2	2015-01-10 20:33:38	1	1.80	-73.963	40.803	-73.952	40.824
3	2015-01-10 20:33:39	1	0.50	-74.009	40.714	-74.004	40.720
4	2015-01-10 20:33:39	1	3.00	-73.971	40.762	-74.004	40.743

pickup_latitude	dropoff_longitude	dropoff_latitude	payment_type	fare_amount	tip_amount	total_amount	duration
40.750	-73.975	40.751	1	12.0	3.25	17.05	1083.0
40.724	-73.994	40.759	1	14.5	2.00	17.80	1190.0
40.803	-73.952	40.824	2	9.5	0.00	10.80	603.0
40.714	-74.004	40.720	2	3.5	0.00	4.80	112.0
40.762	-74.004	40.743	2	15.0	0.00	16.30	1159.0

Data

- **Raw Data for Weather:**

Time		Temp	Weather	Wind		Humidity	Barometer	Visibility
10:51 am		45 °F	Sunny.	13 mph	↘	33%	29.72 "Hg	10 mi
11:51 am		45 °F	Sunny.	14 mph	→	31%	29.75 "Hg	10 mi
12:51 pm		46 °F	Sunny.	15 mph	↘	30%	29.77 "Hg	10 mi

- **Weather Crawler:**

- Data Cleansing (repeating/missing time, Temp, weather, etc.)

1	Time	Temp	Weather	Wind	Humidity	Barometer	Visibility
2	01 00:51	28	Clear	10	43	30.14	10
3	01 01:51	28	Clear	9	47	30.14	10
4	01 02:51	27	Clear	7	46	30.14	10
5	01 03:51	27	Clear	3	42	30.12	10

Data

- **Pickup Number Prediction:**

- Feature extraction: pickup passenger(Taxi data);
Temperature, Wind, Humidity, Barometer, Visibility (Weather Data):
- Combine pickup passenger number/hr and weather into one dataframe

	Temp	Wind	Humidity	Barometer	Visibility	Time=0	Time=1	Time=10	Time=11	Time=12	...
0	0.400000	0.290323	0.432099	0.540541	1.0	1.0	0.0	0.0	0.0	0.0	...
1	0.363636	0.193548	0.481481	0.549550	1.0	0.0	1.0	0.0	0.0	0.0	...
2	0.381818	0.258065	0.456790	0.567568	1.0	0.0	0.0	0.0	0.0	0.0	...
3	0.363636	0.161290	0.481481	0.612613	1.0	0.0	0.0	0.0	0.0	0.0	...
4	0.290909	0.096774	0.567901	0.639640	1.0	0.0	0.0	0.0	0.0	0.0	...

Data

- **Time series forecast**

- Feature extraction: tpep_pickup_datetime, passenger_count (taxi data)
- Read a particular hour and number of passengers travelling in that hour as time series to forecast the hourly passenger count in New York.
- Due to limited time, we did not apply time series for each specific location(e.g. Manhattan, Bronx, Brooklyn, Queens, etc.). We are thinking about to set different size of rectangles boundaries and apply time series to each region.

Data

- **Cost prediction for a specified trip:**
 - Feature extraction: trip_distance, duration, total_amount (taxi data);
 - Temp, Weather, Wind, Visibility, Weather=clear, etc. (weather data).
 - Combine taxi data and weather data into one dataframe and used for prediction

	trip_distance	duration	Temp	Wind	Visibility	Weather=clear	Weather=cloud	Weather=fog	Weather=heavy rain
Time									
16	2.07	490.0	48	17	10	0.0	1.0	0.0	0.0
16	1.00	494.0	68	5	7	1.0	0.0	0.0	0.0
17	0.79	83037.0	68	6	7	0.0	1.0	0.0	0.0
21	1.04	448.0	57	3	9	0.0	1.0	0.0	0.0
7	0.90	413.0	63	7	10	0.0	1.0	0.0	0.0

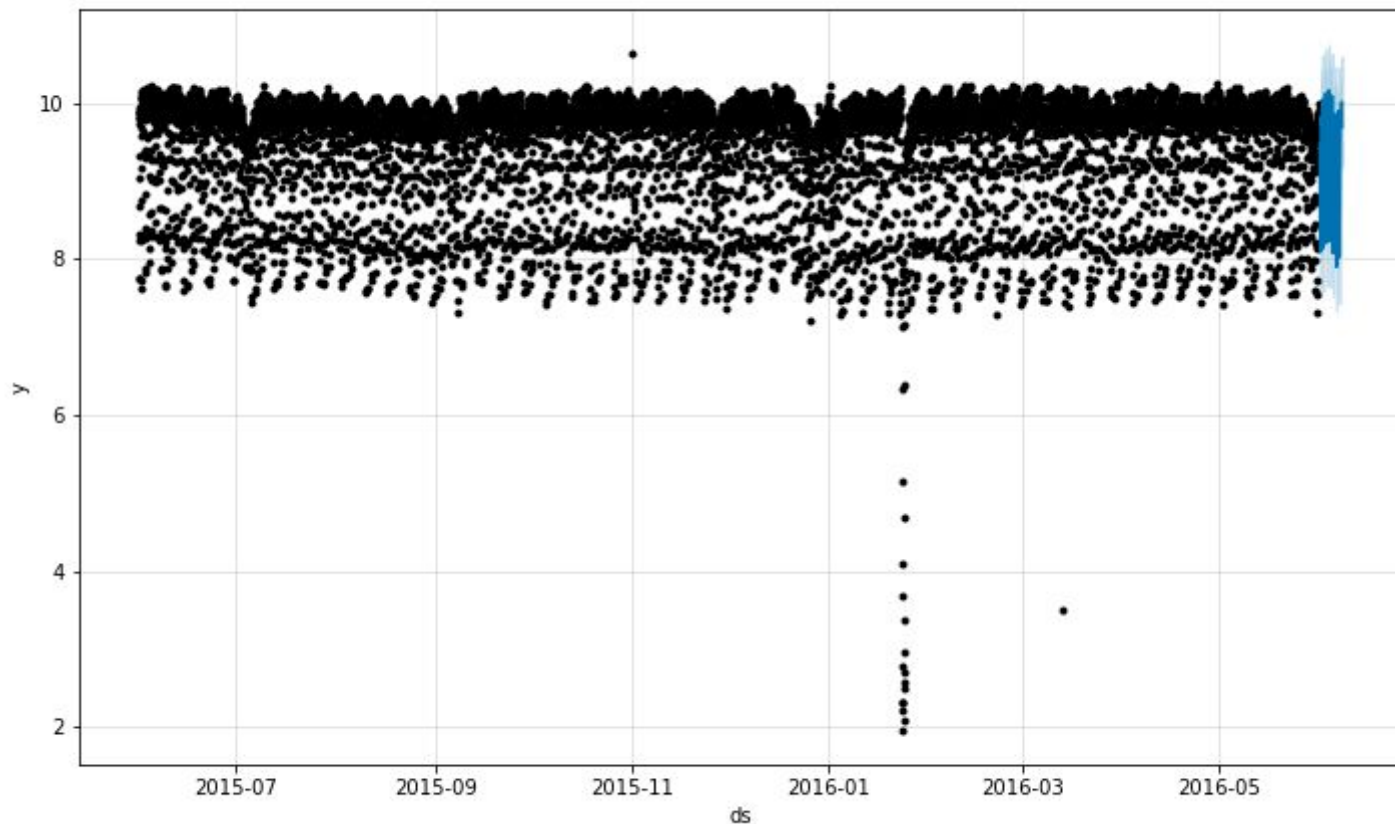
Methodologies

Pickup Number Prediction

- **Data:** Using hourly data from 2016-03 to 2016-04 to train and predict models. The last week of 2016-04 is the test set and the rest is the training set
- **Daily Experience:** Building models for weekdays and weekends separately
- **Ten Regression Models:** Estimating the relationships among variables
- **GridSearchCV:** Tuning the hyper-parameters of an estimator
- **Learning Curve:** Finding out how much we can benefit from adding more training data and whether the estimator suffers more from a variance error or a bias error

Methodologies

Time series predicted total pickup number during the first week of June with historical data

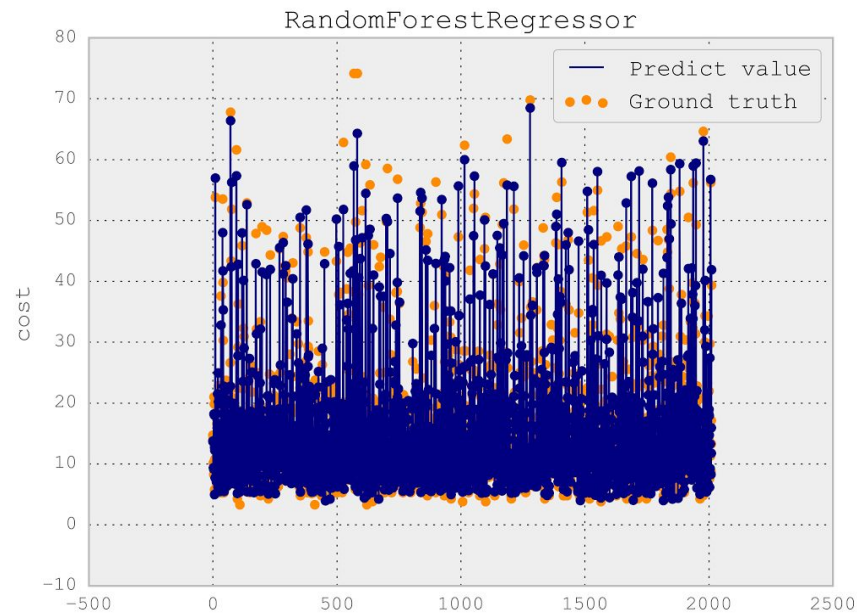
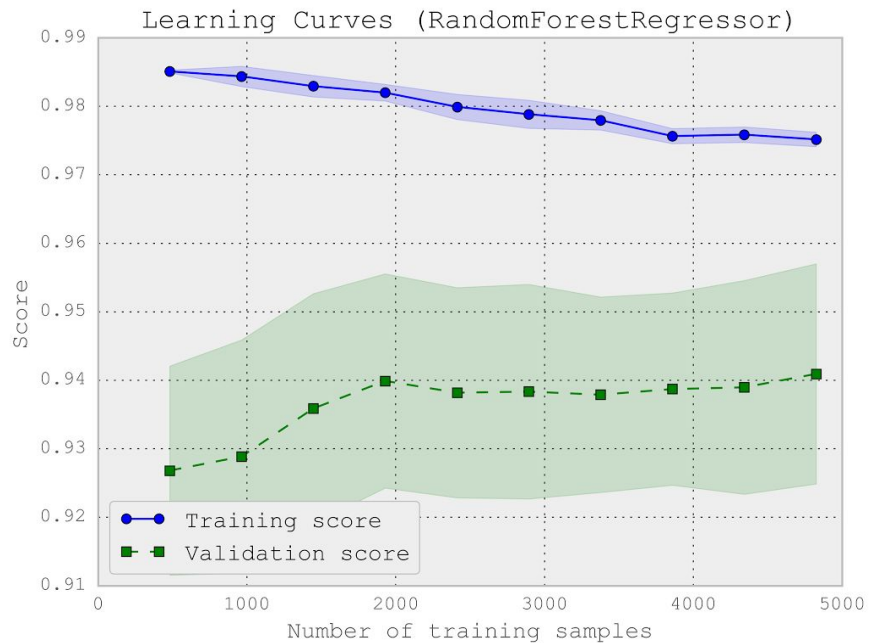


Methodologies

Trip Cost Prediction

Model	Implementation	r^2 _score
SVR RBF kernel	SVR(kernel="rbf", C=100, gamma=0.01}	0.762
Gradient Boosting Regression	GradientBoostingRegressor(learning_rate=0.01, n_estimators=900)	0.9524
Kernel Ridge Regression	KernelRidge(kernel="rbf", alpha=0.01, gamma=0.01)	0.4185
DecisionTreeRegressor	DecisionTreeRegressor(max_depth=6)	0.9433
KNeighborsRegressor	KNeighborsRegressor(n_neighbors=4)	0.8165
RandomForestRegressor	RandomForestRegressor(n_estimators=500, max_depth=8)	0.955
AdaBoostRegressor	AdaBoostRegressor(n_estimators=20)	0.9262

Methodologies



Predict value and ground truth for trip cost

Methodologies

iOS Application for Displaying Cost
Based on Google Maps SDK for iOS



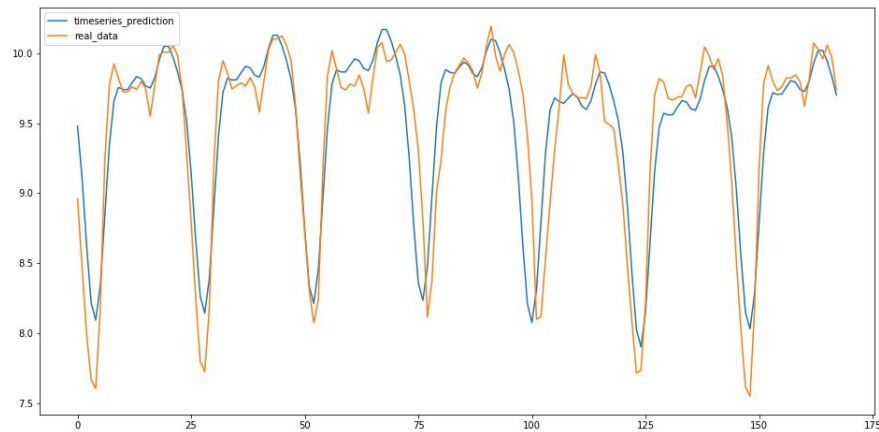
```
connection established
('160.39.38.105', 54274)
client_data: 40.6413111,-73.7781391|40.758895,-73.985131
[ 1.81000000e+01  2.59900000e+03  4.80000000e+01  8.00000000e+00
 0.00000000e+00  0.00000000e+00  1.00000000e+00  0.00000000e+00
 0.00000000e+00  0.00000000e+00  0.00000000e+00]
/root/anaconda2/lib/python2.7/site-packages/sklearn/utils/validation.py:386: Deprecat
ionWarning: Passing 1d arrays as data is deprecated in 0.17 and will raise ValueError
in 0.19. Reshape your data either using X.reshape(-1, 1) if your data has a single fe
ature or X.reshape(1, -1) if it contains a single sample.
  DeprecationWarning)
43.32
58.85
connection closed
```

Performance Measures

1. Prediction of Total Pickup Number Per Hour
2. Prediction of Pickup Number Per Hour on Specific Coordinate
3. Prediction of Cost Per Trip
4. iOS Application

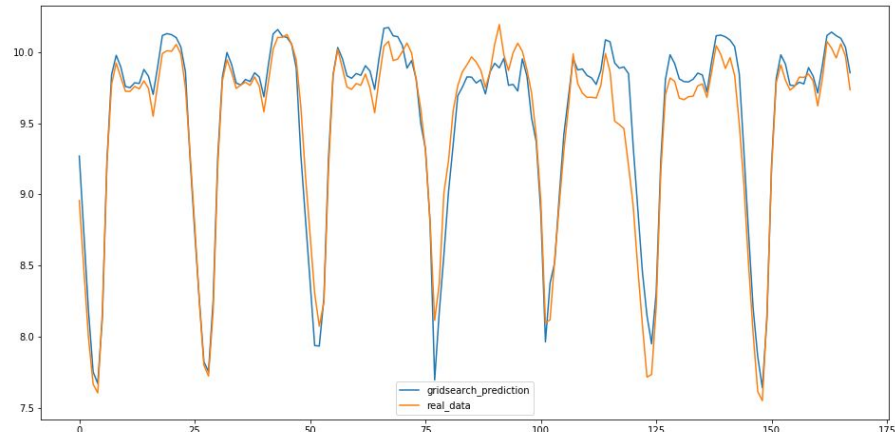
Performance Measures

Prediction of Total Pickup Number Per Hour



Time Series

Pros: Fast, 2-dimensional
Cons: Less accurate

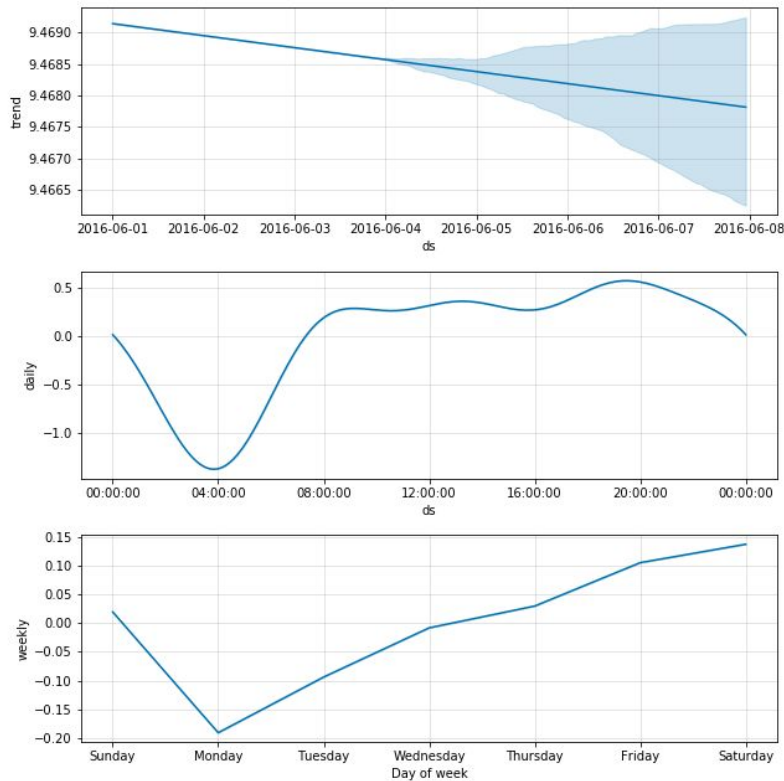


Regression Model

Pros: Accurate, flexible
Cons: Slow, large data

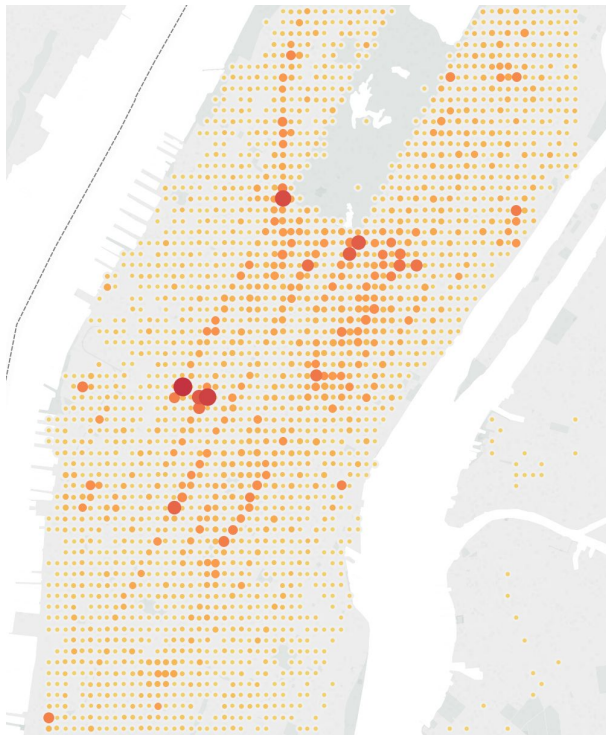
Performance Measures

Time Series Performance

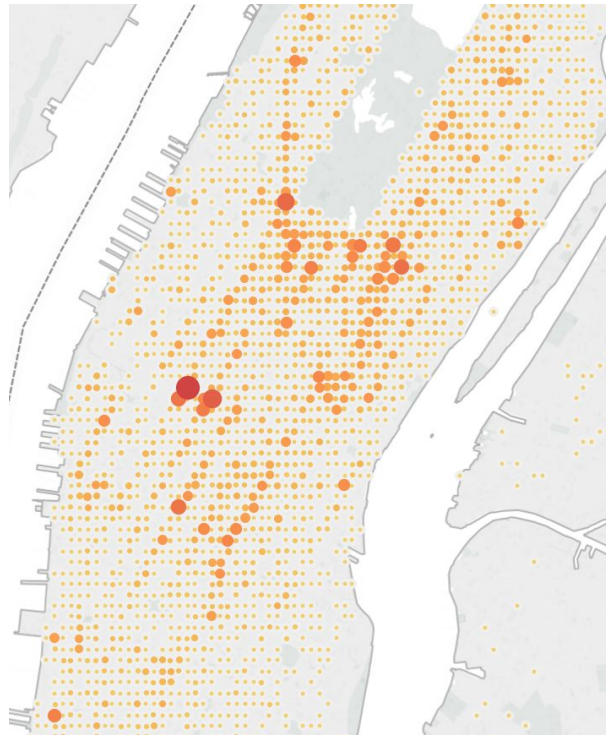


Performance Measures

Prediction of Pickup Number Per Hour on Specific Coordinate



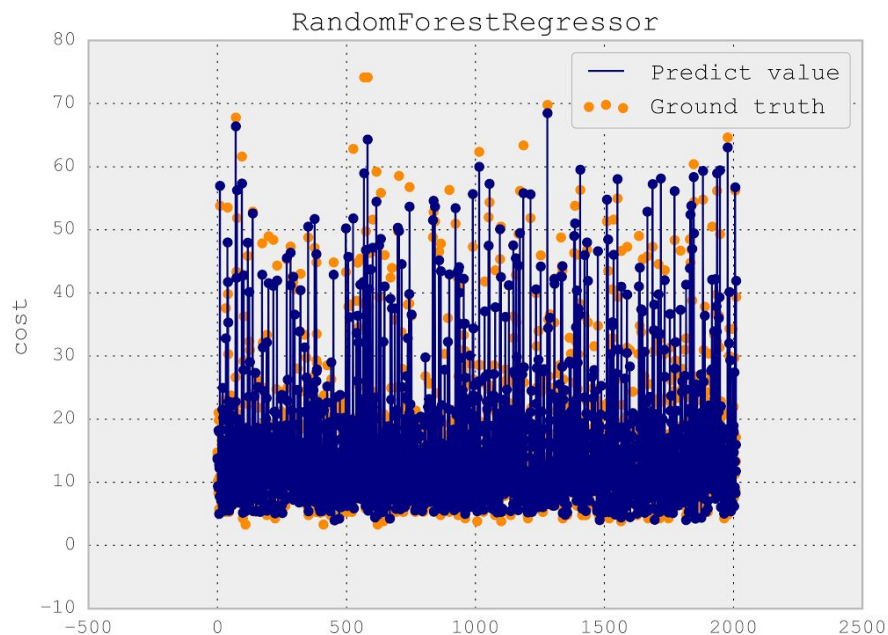
Predicted 18:00-19:00 on 05/03/2016



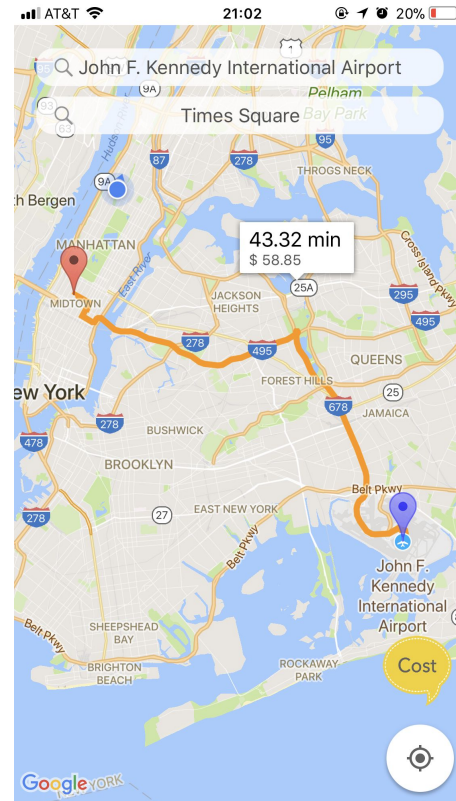
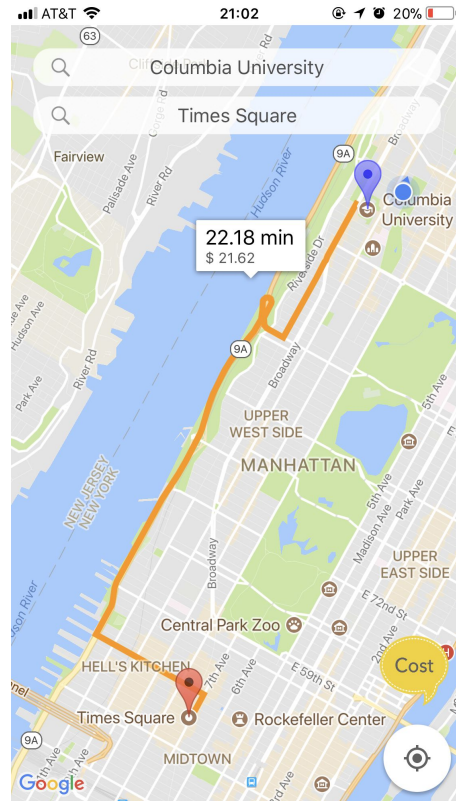
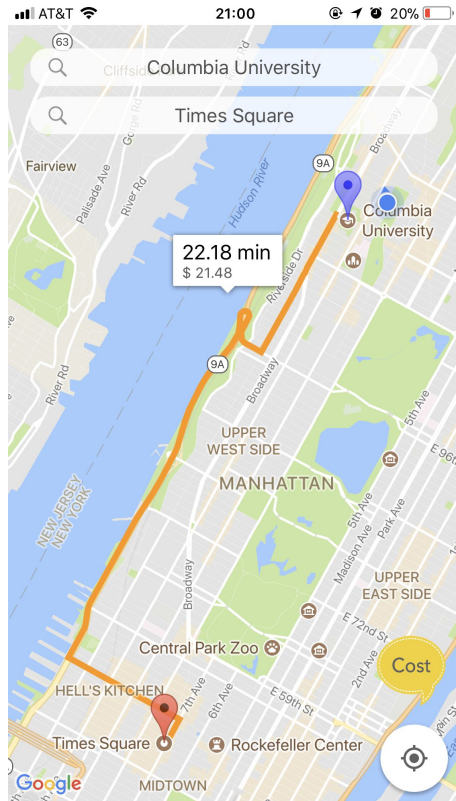
Real 18:00-19:00 on 05/03/2016

Performance Measures

Prediction of Cost Per Trip



Performance Measures iOS Application



What we like the least and the best about the project

- **Least:**

Computing power: Huge dataset and slow computer processing (AWS)

Time constraint: Due to limited time, we can only apply time series method to aggregate hours for a broad region(Manhattan) instead of hours for each specific location.

Solution: Plan ahead subproject time and consider the time series method in advance. Pick a specific region from dataset and apply time series method to each region individually.

Insufficient data information: No customer actual request time; otherwise can have more efficient scheduling

What we like the least and the best about the project

- **Best:**

Successful prediction: Successfully predicted the prices of the Yellow Cab trips

Learning Opportunity: Opportunity to self-learned the basic process of data science project for the real world problems

Grid Search: Professor's package, Grid Search, is applicable to our project

Monetization: Potential monetization opportunity to create our own startup, Mega Taxi

Conclusion

Summary:

- **Data used:** NYC yellow taxi data, extracted useful columns, reduced to half size, incorporate weather data
- **Models adopted:** Time Series, Kernel Ridge Regression, Random Forest Regression
- **Results predicted:** Total hourly pickup number, hourly pickup number by coordinate, the cost of each trip
- **Feature built:** Mega-Taxi iOS App