New York City Uber Pickup Analysis and Prediction

Chuan Chen, Yafu Ruan, Shuwen Shen, Yihang Zhang

I. INTRODUCTION

The demand for ride-sharing services in NYC can vary in different areas, time periods, or under different weather conditions. Based on the Kaggle dataset of NYC Uber Pickups for 6 months starting from January 2015, we aimed to analyze and predict the demand (measured and represented by the number of pickups) and to analyze the relationships between the number of pickups and variables including weather, location, and several time factors. We try to produce results that could provide insights to help ride-sharing companies like Uber improve the user experience through optimizations in planning.

II. DATA OVERVIEW

A. Dataset Description

Our dataset "NYC Uber Pickups with Weather and Holidays" comes from Kaggle [1]. The dataset consists of the hourly data collected from 01/01/2015 to 30/06/2015.

Overall, the dataset contains 29,101 instances, each of which has the following 13 variables:

TABLE I
DATA OVERVIEW

feature	definition
pickup_dt	Time period of the observations
borough	NYC's borough
pickups	Number of pickups for the period
spd	Wind speed in miles/hour
vsb	Visibility in Miles to nearest tenth
temp	temperature in Fahrenheit
dewp	Dew point in Fahrenheit
slp	Sea level pressure
pcp01	1-hour liquid precipitation
pcp06	6-hour liquid precipitation
pcp24	24-hour liquid precipitation
sd	Snow depth in inches
hday	Being a holiday (Y) or not (N)

B. Exploratory Data Analysis

We applied several data visualization techniques trying to discover how different features would affect Uber ridership.

From the bar plot in terms of the number of pickups and borough, we could see that Manhattan, not surprisingly, accounts for the largest number of Uber pickups of any borough, and EWR produces the smallest. Weather events, for example, the snowfall, would definitely have significant impact on daily Uber ridership. According to the bar plot of snow depth variable, it appears that when snow depth is

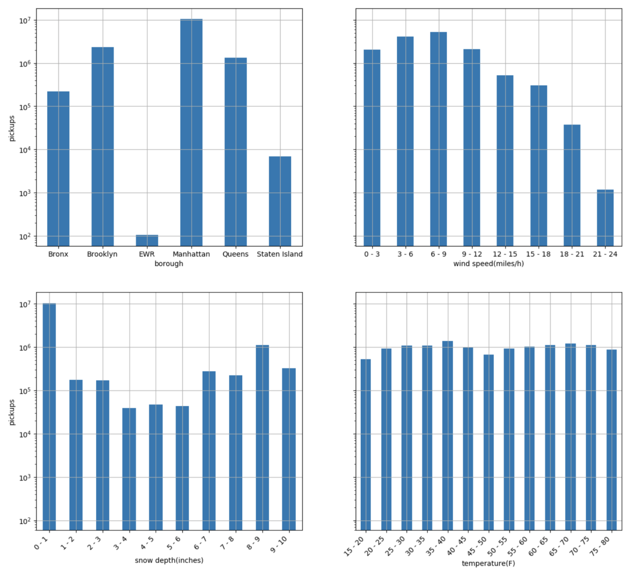


Fig. 1. Bar plot of the relationship between 4 features to pickup amount

greater than 6 inches, the amount of Uber pickups would significantly increase.

We also constructed a heat map to show the pattern and magnitude of the number NYC Uber pickups in terms of month and hour. on every night after 5pm from January to

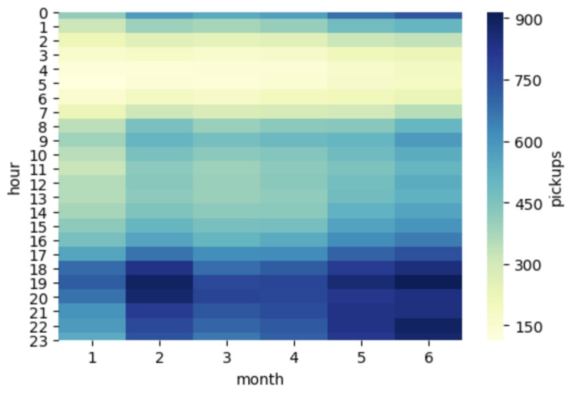


Fig. 2. Heatmap generated by Python Seaborn reflecting Uber pickups values during a day over 6 months.

June, among all 6 months, Uber provides more trips than any other time periods and reaches its peak during the whole day. This is probably because people are more likely to feel tired after a long day of work, so that they tend to rely on Uber pickup system to rush home or enjoy dinner time with friends. Also, we found that the Uber ridership in February and June is relatively high compared to other months. This could be explained by the inclement weather temperature during February and June.

III. Data Preprocessing

A. Missing Values

There is one type of missing values ("NaN") in one of the columns ("borough") in our dataset. We checked that 3043 out of 29101 (total) records had "NaN". Since this represents a small proportion of the total data available, we decided to drop the records containing missing values.

B. String Values

There are two columns with string values in our dataset - "hday" and "borough". We transformed both to numerical columns. Specifically, "Hday" has values "Y" (if it is a holiday) and "N" (if it is not a holiday). We transformed the values into 1 (for "Y") and 0 (for "N")

C. Dummy Variables

For regression models, we use one-hot encoding to convert borough feature into six dummy variables with names "Bronx", "Brooklyn", "EWR", "Manhattan", "Queens", and "Staten Island". A record has a value of "1" if it belongs to the borough that the dummy variable represents and "0" if not.

D. Date Features

The original date feature is not in pandas date time format. Therefore, we converted it into different features based on the demands of different models, which will be specified in the Modeling section.

E. Feature Normalization

According to that most of the features in our dataset are having greatly different scale, we applied normalization on numerical features so as to fit the model better.

IV. Modeling

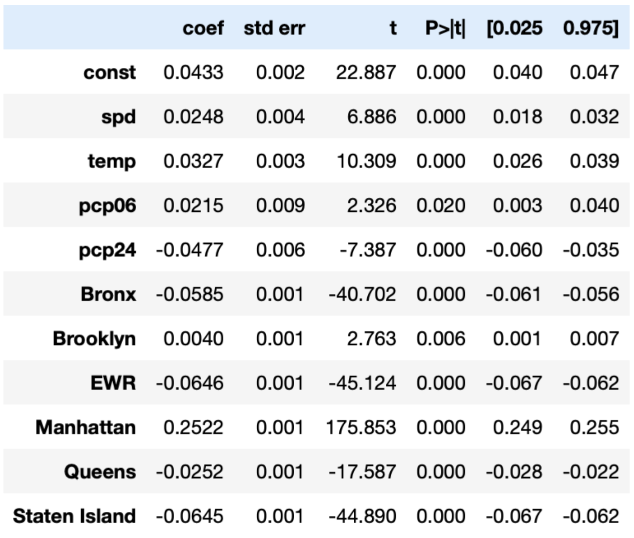
A. Linear Regression Model

As our baseline model, we expected to extract the linear relationship between features and target variable so as to understand how pickup amount is influenced by those features.

To process the date feature, we converted it into three categorical features, day, month and range. Day refers to the exact date within month, from 1 to 31. Month refers to the exact month of the record, from 1 to 6 since the dataset only cover January to June. And range refers to 8 three-hour ranges(e.g. 1 means 1 am to 3 am, 2 means 4am to 6 am).

After we splited the dataset, we fitted the model and checked the multicollinearity. The R-squared was 0.701 and the RMSE was 0.08346, which was not bad. While looking at the detailed summary, we noticed that several features were statistically insignificant and some pairs of features had serious multicollinearity problem, such as dew point and temperature. Therefore, after feature selection, the model got improved a little and all kept features were now statistically significant. However, by looking at coefficients, we found it was difficult to explain the influence of date features on the target variable. The reason might be that the model would take date features as numerical features instead of categorical features since they were highly ordinal.

In this case, we simply excluded the date features for linear regression to see how the rest features affect the target variables without considering time. The R square went down to 0.660 since we decrease the feature amount but the F Statistics went up from 2190 to 3660, which indicates a better fit and model explanation.



	coef	std err	t	P>|t|	[0.025	0.975]
const	0.0433	0.002	22.887	0.000	0.040	0.047
spd	0.0248	0.004	6.886	0.000	0.018	0.032
temp	0.0327	0.003	10.309	0.000	0.026	0.039
pcp06	0.0215	0.009	2.326	0.020	0.003	0.040
pcp24	-0.0477	0.006	-7.387	0.000	-0.060	-0.035
Bronx	-0.0585	0.001	-40.702	0.000	-0.061	-0.056
Brooklyn	0.0040	0.001	2.763	0.006	0.001	0.007
EWR	-0.0646	0.001	-45.124	0.000	-0.067	-0.062
Manhattan	0.2522	0.001	175.853	0.000	0.249	0.255
Queens	-0.0252	0.001	-17.587	0.000	-0.028	-0.022
Staten Island	-0.0645	0.001	-44.890	0.000	-0.067	-0.062

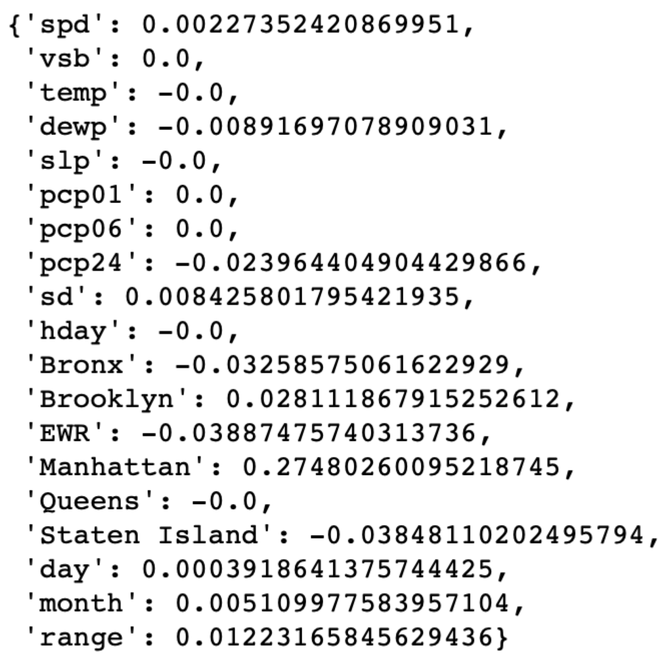
Fig. 3. Coefficients of Linear Regression

The current model seems better, with a higher F statistics and lower RMSE. It is acceptable for R Square experiencing a decrease because less features are included. Almost all of the features are statistically meaningful except Queens but we need to keep it. By looking at the coefficients, we can see features positively influencing the pickup amounts are: windspeed, temperature, Brooklyn and Manhattan, meaning that with higher windspeed, temperature, or when place is in Brooklyn and Manhattan, we expect higher amount of pickups. Vice Versa with the rest features.

B. Elastic Net

In the linear regression model, we manually filtered the features when dealing with multicollinearity. Therefore, we wanted to applied Elastic Net Model by adding regularization term to automatically shrink and penalize features. To determine the parameters, we applied MPI to accelerate the grid searching process. As the result, when alpha is 0.0011 and l1 ratio is 0.1, we get the best model with 0.7246 as R2 and 0.0836 as RMSE.

According to the output, we got how the features in this model linearly influence the target feature. Some of the features' coefficients have been shrinked to 0, indicating they are filtered out because of multicollinearity. Most of them are



```
{'spd': 0.00227352420869951,
 'vsb': 0.0,
 'temp': -0.0,
 'dewp': -0.00891697078909031,
 'slp': -0.0,
 'pcp01': 0.0,
 'pcp06': 0.0,
 'pcp24': -0.023964404904429866,
 'sd': 0.008425801795421935,
 'hday': -0.0,
 'Bronx': -0.03258575061622929,
 'Brooklyn': 0.028111867915252612,
 'EWR': -0.03887475740313736,
 'Manhattan': 0.27480260095218745,
 'Queens': -0.0,
 'Staten Island': -0.03848110202495794,
 'day': 0.0003918641375744425,
 'month': 0.005109977583957104,
 'range': 0.01223165845629436}
```

Fig. 4. Coefficients of Elastic Net Model

exactly the features have been dropped in Linear Regression Model.

C. ARIMA

Since our Uber pickups data is collected hourly in sequential order, it is a time series in nature. Thus, it would make sense to try analyzing the data using time series models, such as the ARIMA model. The downside of evaluating the dataset using the ARIMA model is that all features beside time are discarded, and their collection efforts are wasted. However, the upside is that it can capture the strong time dependency apparent in the data, something that ordinary machine learning models can not do. Since the same time point repeats 6 times in our dataset, one for each borough, we are faced with 2 options to prepare our dataset for time series analysis. We can either take the average pickup numbers across all 6 boroughs for each time point, or we can split the dataset into 6 sub-dataset and perform time series analysis on each borough-specific data frames. Since each borough differs significantly from each other, averaging them all together would cause us to lose insight into the area-specific effects on the pickup numbers. During deployment, allowing Uber drivers and consumers to see the prediction for the specific borough they are located in will certainly be more helpful then providing them with the average pickup numbers across the entire City. Thus, we have decided to split our data into 6 borough-specific data sets. Then, all features beside time are discarded, and the data was normalized. From the decomposition for the Bronx dataset, as shown in figure 5, we can clearly observe a daily periodicity pattern, and the same seasonality is apparent in all other data sets. The ARIMA model was fitted on each of the 6 data sets, and a total of 64 different parameters was tested on each

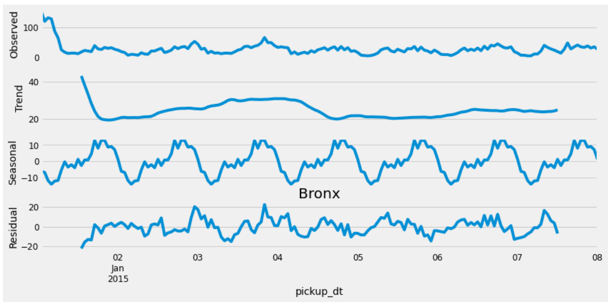


Fig. 5. times-series decomposition for the Bronx dataset

fit, setting the periodicity to be 24. The best parameter for each fit was selected based on the lowest AIC score, which is an indicator that balances goodness of fit and model complexity. Afterwards, the models were refitted on each of the 6 data sets using their corresponding best parameters. The result of each fit is shown in appendix A.1. Figure 6 shows that the residual after the ARIMA fit is near normally distributed, which suggest that they are random noise, and we have extracted most of the pattern from our data. The

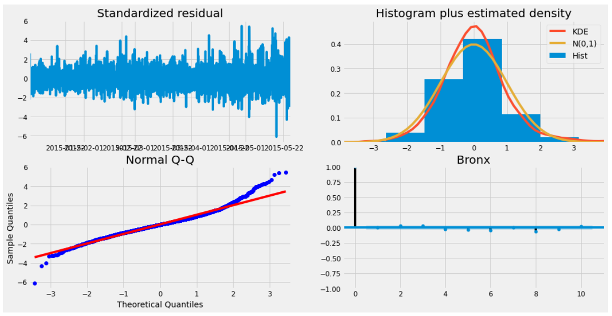


Fig. 6. residuals for the Bronx model

residuals for the rest of the models are shown in appendix A.2. All model residuals are nearly normally distributed. An in-sample prediction was performed for all data sets for a 15-day period. Results are shown in appendix A.3. The average prediction RMSE is 0.0690. An out of sample forecast was conducted for a 36-day period and compared with the validation set. Results are shown in appendix A.4. The final average forecast RMSE is 0.1645.

D. Neural Network

We also uses Neural Network as to deal with the nonlinear relationship between the features and target variables. Here mean square error was used as the objective function.

Here we also introduce MPI to speed up the gridsearching for hyperparameters. For this problem we find out that a deep network may not be a good choice, so we design a simple only two hidden layers. Relu is chosen as the activation function and the slover is set to be adam after parameter tuning.

The network converges within 50 epoch and and MSE on training set reaches 0.0024 as shown in figure 5. We also introduce the early stopping technique to avoid overfitting.

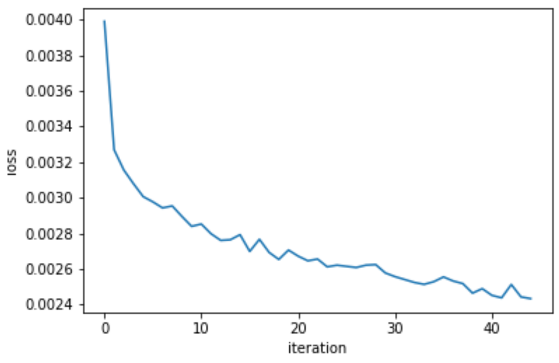


Fig. 7. Training loss on neural network

Finally the RMSE on the validation set is 0.0703, which is slightly better than linear models. However this model is hard to explain and it's difficult to find the relationship between features and variables.

V. Performance Improving

A. *line_profiler*

The line_profiler shows how much time is spent one each function. It helps us understand which line actually causes the program slow and it may be obvious enough. In our project, we first use line_profiler to locate the part that need to be optimized and then apply several techniques.

B. *Itertools*

The itertools provides a number of iterators that is fast and efficient compared with operation on list. The reason is that itertools do not produce the data until it is needed. Since it reduce the swag between data, it speed up the program in large dataset. We use itertools in gridsearching part to find all combinations of hyperparameters. It shows that using list operations will take more time. While setting up the hyperparameter search grids for the ARIMA model, a for-loop took 0.0052s, while an itertools method only took 0.0045s, which is a 13.46% performance increase.

C. *Parallel Programming*

We uses Message Passing Interface(MPI)[2] for parallel computing. It is a standard portable messaging system designed for parallel system. MPI is used to hyperparameter gridsearching since it could be easily paralyzed. We will find all combinations of the parameters and assign them to different processors. For each processor, it will build model independently. Eventually, the best combination of parameters among all processor will be chosen. In the table below shows the difference of using MPI vs not for NN parameter tunning . It indicates that MPI saves 36.7% of time.

MPI is also applied in the parameter tuning of Net Elastic Model. It speeds up the process by 38%, from 0.2492s to 0.1545.

TABLE II
Comparision of MPI and Normal For-loop in Neural Network

MPI	For-loop
258s	408s

D. *Python Concurrency*

Since parameter tuning is a CPU-bound task, we have also tried to use multiprocessors to search for the best parameters for our ARIMA model. It can easily divide up the task of fitting all possible parameters between the different processors and carry out multiple tasks simultaneously. We have divided the search grid into 4 chunks, one for each CPU. Due to the nature of multiprocessing, dividing into more chunks will not only be unhelpful, but might even decrease performance. We have tried to divide the search grid into 8 chunks, but the search time was almost doubled. This is understandable since more chunks do not mean more processes in motion, just more processes in waiting. We obtained the lowest AIC score from each of the 4 process. The minimum of those 4 AIC scores then returned the final optimal parameters for each model. The overall time improved about 21.22

Since we did not encounter any I/O bound tasks in our project, we have not optimized any performance using threading.

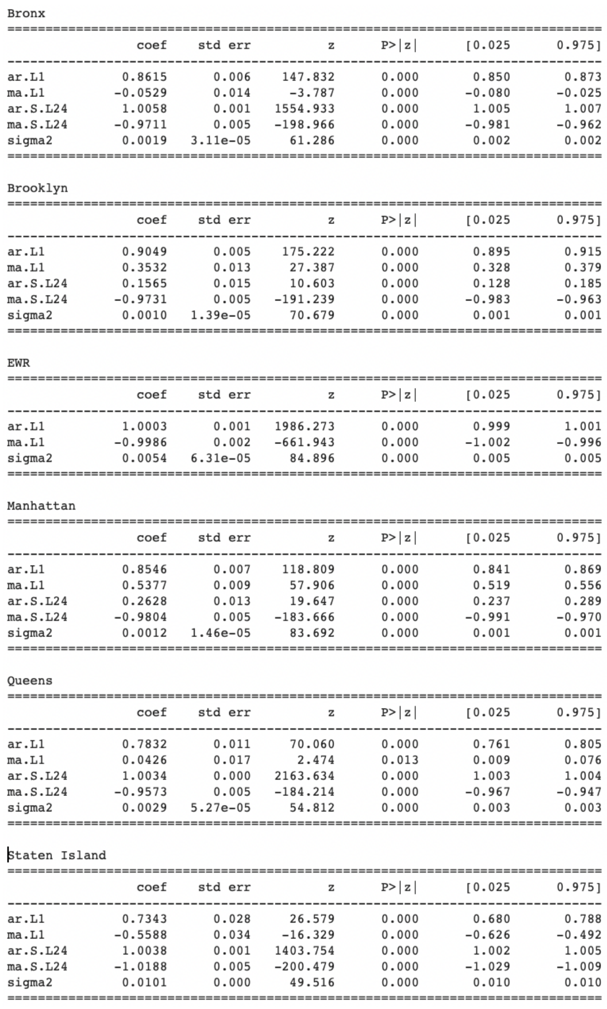
VI. Conclusions

Out of the four models we have tried, Neural Networks yield the lowest RMSE of 0.0703. Ranking the coefficients of the linear regression model and the elastic net model, it seems that the borough locations and daily precipitation effects pickup numbers the most. Through the means of performance optimization, we have increased our efficiency by approximately 27.35% overall. For future work, we may suggest exploring possible additional features to add to the dataset that may help make up for the highly-multicolinear features. If given time, we also suggest trying out more parameters for the ARIMA model. Lastly, since neural networks showed such good performance, trying out additional deep learning models such as RNN and LSTM seems promising.

Appendix

A. *Modeling*

1) ARIMA model results: Best fit ARIMA models for each borough-specific dataset.



```
Bronx
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.8615      0.006    147.832      0.000       0.850       0.873
ma.L1         -0.0529      0.014     -3.787      0.000      -0.080      -0.025
ar.S.L24       1.0058      0.001   1554.933      0.000       1.005       1.007
ma.S.L24      -0.9711      0.005   -198.966      0.000      -0.981      -0.962
sigma2         0.0019   3.11e-05     61.286      0.000       0.002       0.002
==============================================================================

Brooklyn
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.9049      0.005    175.222      0.000       0.895       0.915
ma.L1          0.3532      0.013     27.387      0.000       0.328       0.379
ar.S.L24       0.1565      0.015     10.603      0.000       0.128       0.185
ma.S.L24      -0.9731      0.005   -191.239      0.000      -0.983      -0.963
sigma2         0.0010   1.39e-05     70.679      0.000       0.001       0.001
==============================================================================

EWR
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          1.0003      0.001   1986.273      0.000       0.999       1.001
ma.L1         -0.9986      0.002   -661.943      0.000      -1.002      -0.996
sigma2         0.0054   6.31e-05     84.896      0.000       0.005       0.005
==============================================================================

Manhattan
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.8546      0.007    118.809      0.000       0.841       0.869
ma.L1          0.5377      0.009     57.906      0.000       0.519       0.556
ar.S.L24       0.2628      0.013     19.647      0.000       0.237       0.289
ma.S.L24      -0.9804      0.005   -183.666      0.000      -0.991      -0.970
sigma2         0.0012   1.46e-05     83.692      0.000       0.001       0.001
==============================================================================

Queens
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.7832      0.011     70.060      0.000       0.761       0.805
ma.L1          0.0426      0.017      2.474      0.013       0.009       0.076
ar.S.L24       1.0034      0.000   2163.634      0.000       1.003       1.004
ma.S.L24      -0.9573      0.005   -184.214      0.000      -0.967      -0.947
sigma2         0.0029   5.27e-05     54.812      0.000       0.003       0.003
==============================================================================

Staten Island
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.7343      0.028     26.579      0.000       0.680       0.788
ma.L1         -0.5588      0.034    -16.329      0.000      -0.626      -0.492
ar.S.L24       1.0038      0.001   1403.754      0.000       1.002       1.005
ma.S.L24      -1.0188      0.005   -200.479      0.000      -1.029      -1.009
sigma2         0.0101      0.000     49.516      0.000       0.010       0.010
==============================================================================
```

Fig. 8. best fit ARIMA models for each borough dataset

2) *ARIMA model residuals:* Residuals for each ARIMA model.

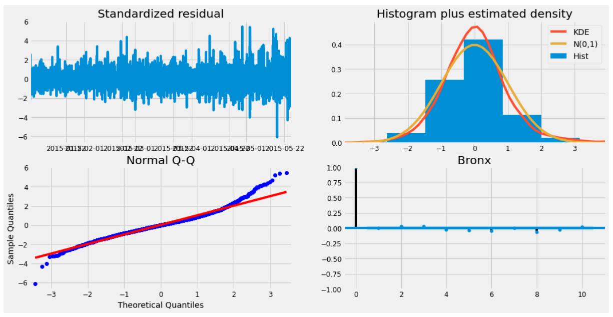


Fig. 9. residuals for the Bronx model

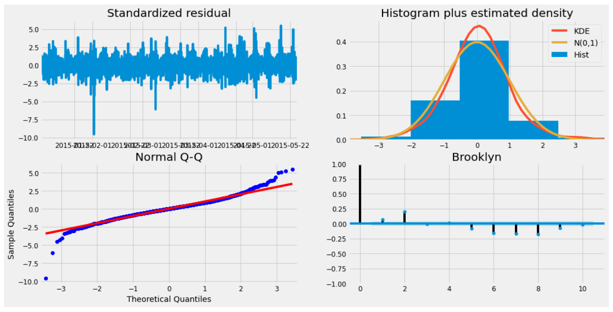


Fig. 10. residuals for the Brooklyn model

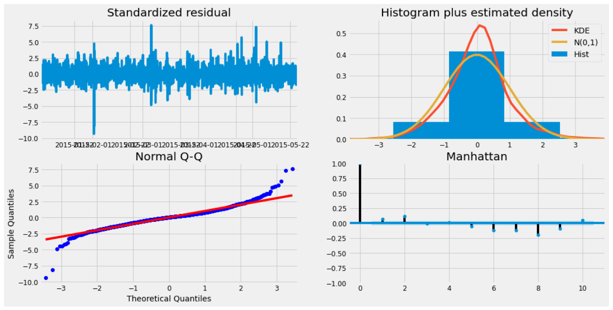


Fig. 11. residuals for the Manhattan model

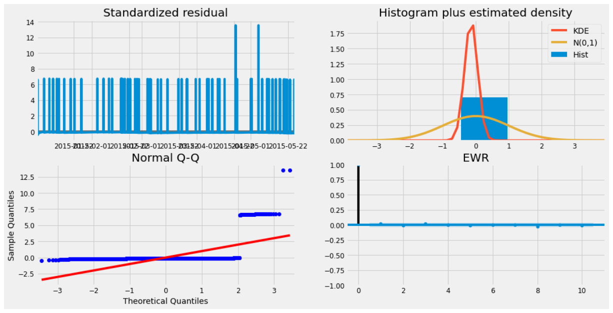


Fig. 12. residuals for the EWR model

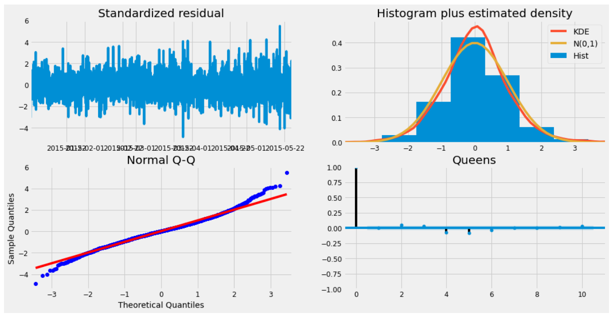


Fig. 13. residuals for the Queens model

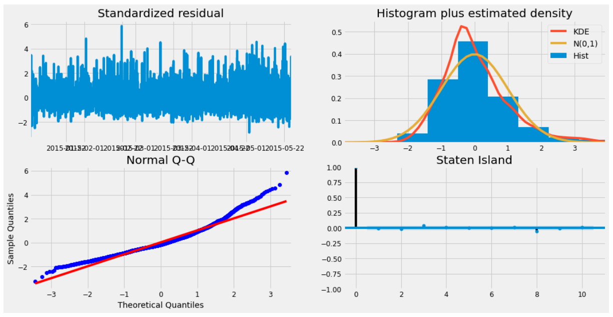


Fig. 14. residuals for the Staten Island model

3) ARIMA model in-sample predictions: In-sample predictions for each ARIMA model.

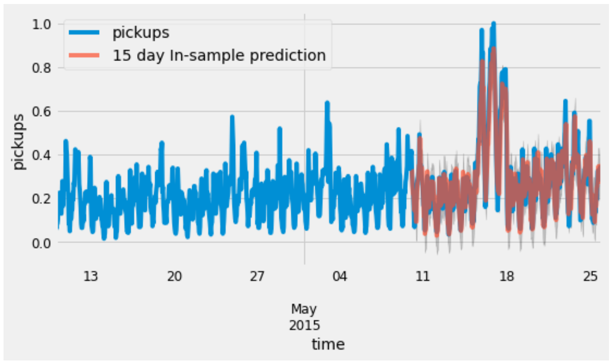


Fig. 15. prediction for the Bronx model

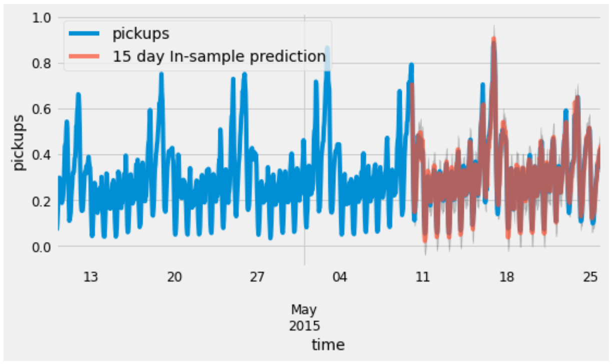


Fig. 16. prediction for the Brooklyn model

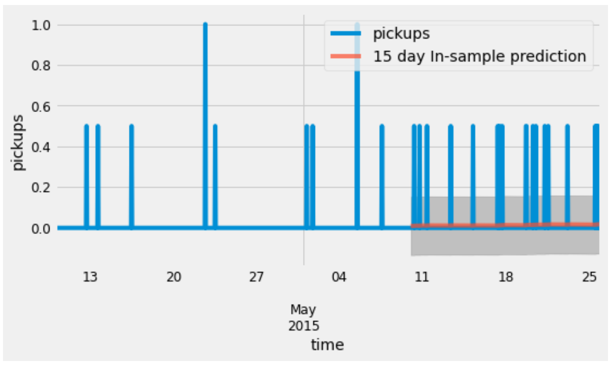


Fig. 17. prediction for the EWR model

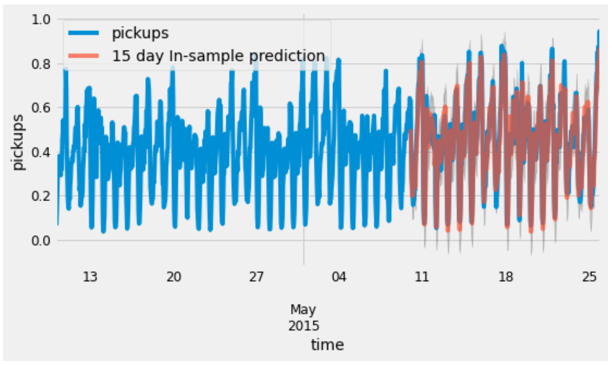


Fig. 18. prediction for the Queens model

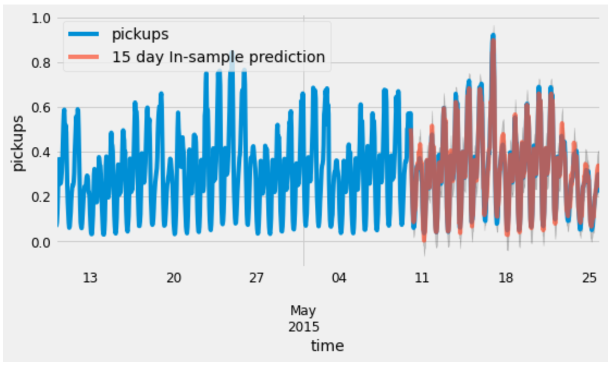


Fig. 19. prediction for the Manhattan model

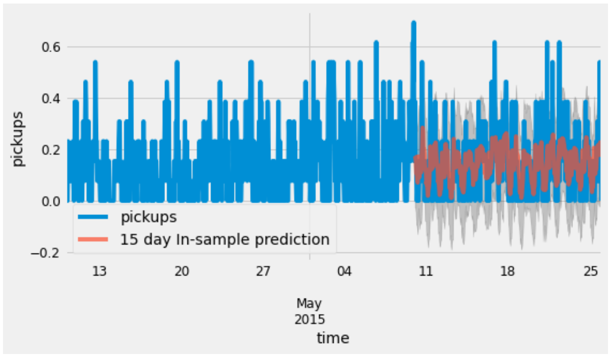


Fig. 20. prediction for the Staten Island model

4) ARIMA model out-of-sample forecast: Out-of-sample forecast for each ARIMA model.

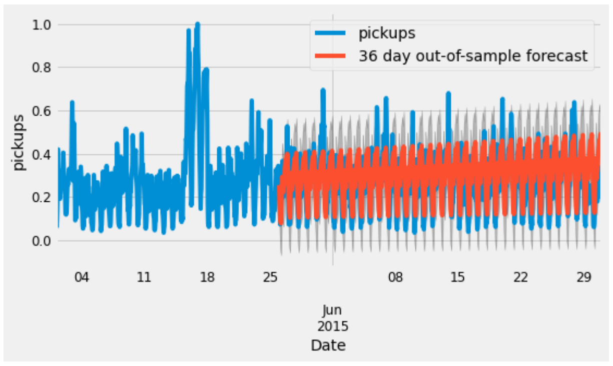


Fig. 21. forecast for the Bronx model

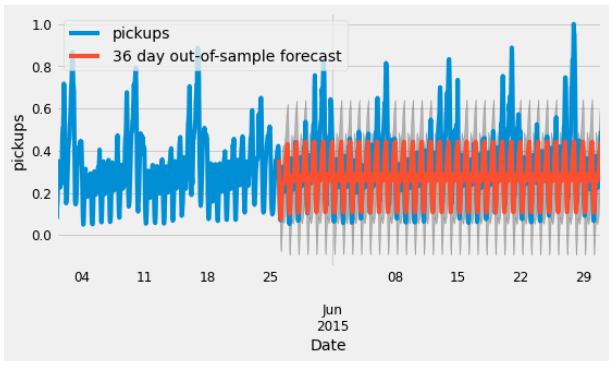


Fig. 22. forecast for the Brooklyn model

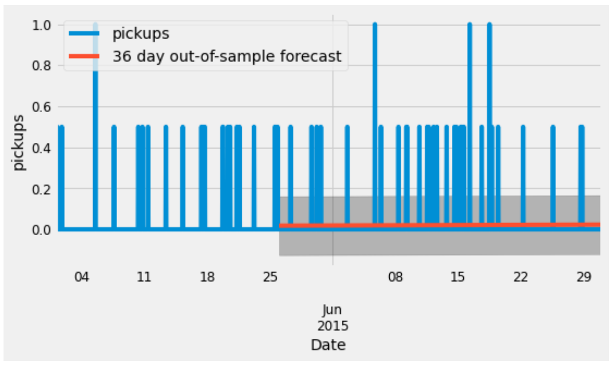


Fig. 23. forecast for the EWR model

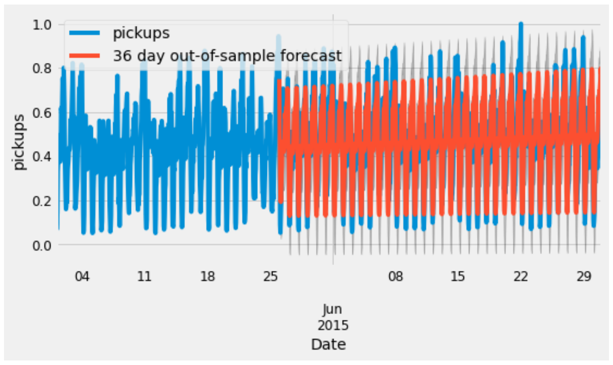


Fig. 24. forecast for the Queens model

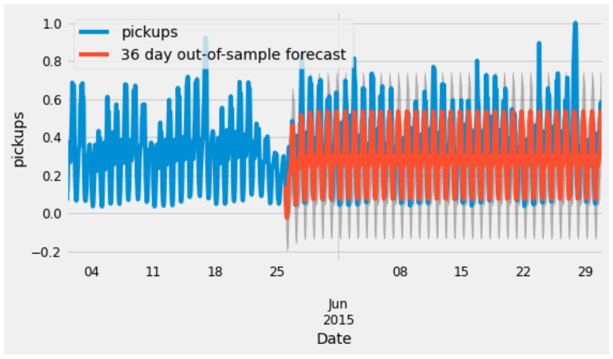


Fig. 25. forecast for the Manhattan model

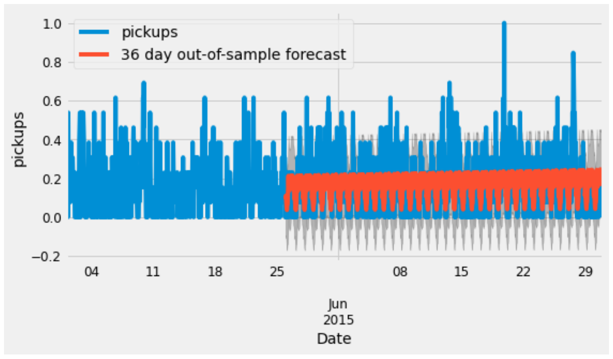


Fig. 26. forecast for the Staten Island model

References

[1] Kaggle, *NYC Uber Pickups with Weather and Holidays Subset of Uber pickup data with weather, borough, and holidays*,https://www.kaggle.com/yannisp/uber-pickups-enriched

[2] MPI for Python, https://mpi4py.readthedocs.io/en/stable/