THE AVAILABILITY OF BIKES: A PREDICTIVE MODEL THAT MAXIMIZES REVENUE FOR CAPITAL BIKESHARE

TEAM COWBOY: ANIRBAN CHOWDHURY, EMILY ZENG, YUCHENG WANG, KEVIN YANG

36-601 PERSPECTIVES IN DATA SCIENCE, 36-611 PROFESSIONAL SKILLS FOR STATISTICIANS

NOVEMBER 30, 2021

AGENDA

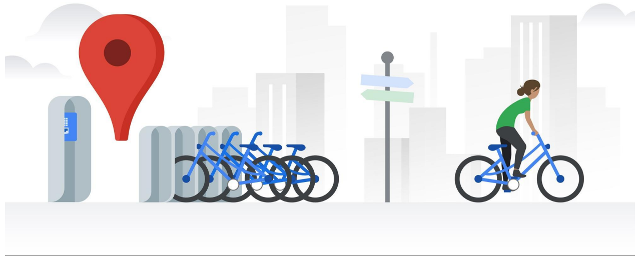
	Slide
Introduction	4
Data	5 - 6
Methodology	7
Accuracy of model	8
Interpretation of model	9
Recommendations	10 - 11
Maintenance of model	12
Limitations	13
Next steps	14
Technical appendix	16 - 24

EXECUTIVE SUMMARY

- Purpose: Capital Bikeshare has tasked Team Cowboy to create a model that can be maintained to predict bike availability and to maximize revenue and / or lower operational costs
- Methods: Team Cowboy employed the use of an extreme gradient boosting model to best predict bike availability
- Recommendations: Our model allows for Capital Bikeshare to predict when / where bikes are the most / least available, thus allowing for more optimal reshuffling of bikes, leading to lower operational costs and maximized revenue overall
- Next steps: Model using different methods such as high-level multivariate time series or use more stations to build the predictive model

INTRODUCTION

- Overall purpose
  - Capital Bikeshare wants to be able to predict bike availability to lower operational costs or maximize revenue
- Scope
  - Build a model that predicts bike availability at a given station for a given time that will allow for scheduling of contractors who reshuffle bikes
  - Document our methodology, maintenance plan, recommendations, and next steps for a phase 2 project



DATA COLLECTION

- Data on bike trips from Capital Bikeshare's website from January 2019 to July 2019
  - Contains information about trip start / end times, start / end stations, and member types
  - Added rows to account for bike reshuffling and used trip information to compute bike availability for every station at every hour
- Additional data on precipitation / temperature per day, whether a day is a holiday / weekend, and max capacity of station (sources for this information found in Technical Appendix)
- Team Cowboy focused on a subset of stations for model building
  - 32 stations in the center of DC, defined as "inner city" stations
  - 28 stations in the surrounding neighborhood north of central DC, defined as "suburban" stations

VARIABLES IN OUR DATASET

Explanatory Variable Category	Variable Name and Descriptions
Station Data	Short name (ID of station), Latitude, Longitude, V (1 = inner city, 0 = suburb), Capacity (max for each station)
Time/Date Data	Hour, Holiday (1 = holiday, 0 = not holiday), Month, Hfact (1 = rush hour (8am - 4pm), 0 = not rush hour)
Precipitation Data	Snow Depth, New Snow, Precipitation (all in inches)
Temperature Data	Maximum, Minimum, Average Temperatures (all per day)

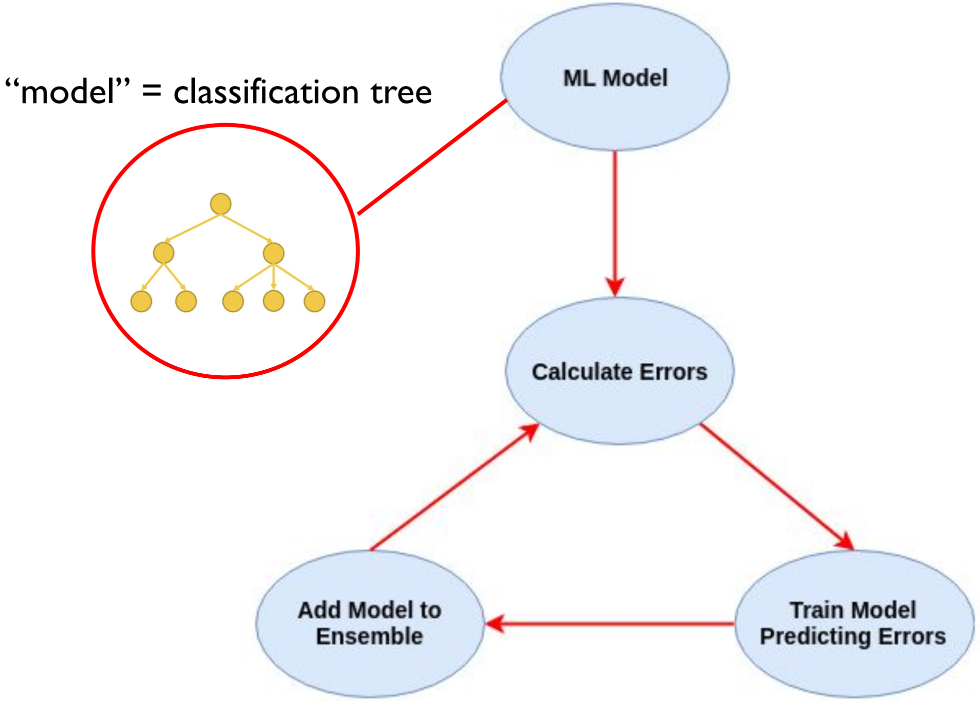
Response Variable	Variable Description
Availability	Availability = Number of available bikes at a station per hour / Capacity, represented as 4 categories: 0: < 25% available 1: 25% - 50% available 2: 50-75% available 3: > 75% available

METHODOLOGY USED TO CREATE MODEL

- Why did we choose an XGBoost model?
  - Fast execution speed
  - Excellent model performance
  - Can be continually updated with new data
- How does it work?
  - Uses multiple separate models to correct the errors of an initial model
  - Add all models' results together to make predictions

NOTE: EVERY MENTION OF MODEL ON THIS SLIDE REPRESENTS A CLASSIFICATION TREE

dmlc XGBoost



Process of XGBoost

INTERPRETATION OF ACCURACY

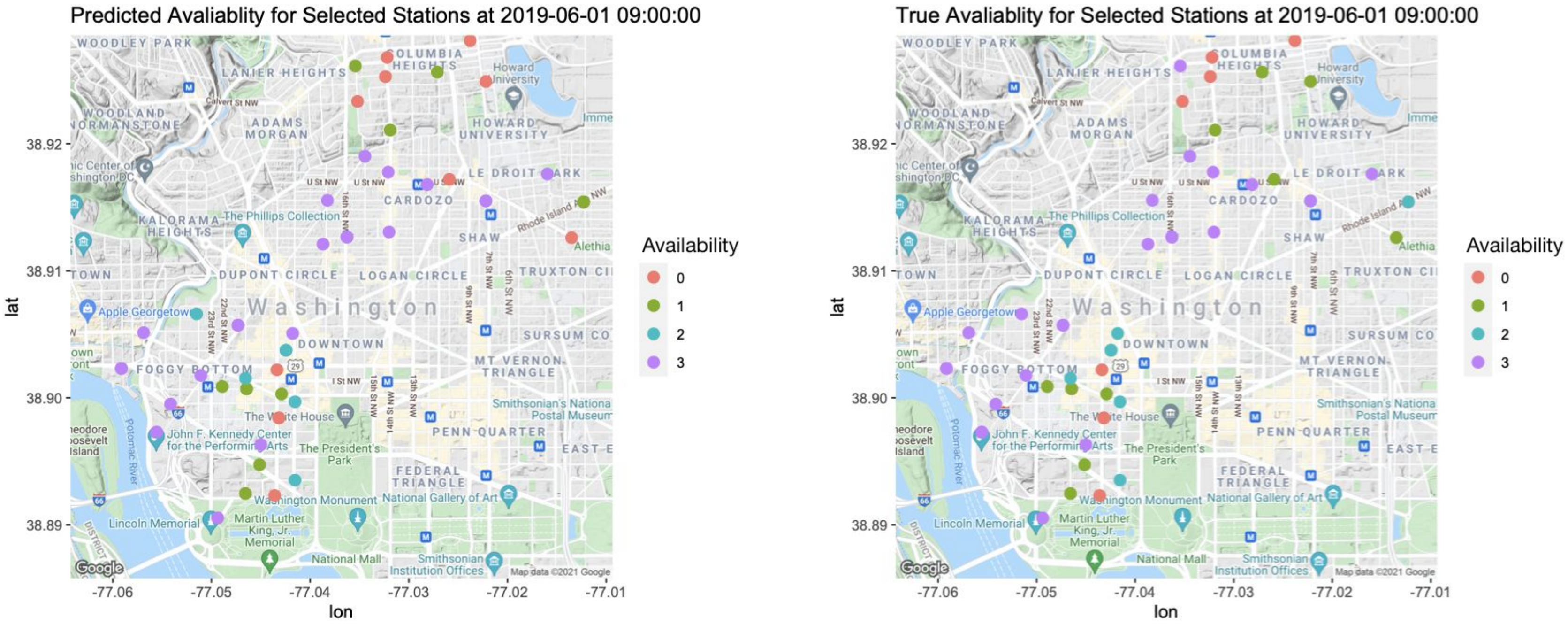
- Accuracy = Number of data points classified correctly / Total amount of data
- Prediction accuracy: 98% accuracy for training data, 86% accuracy for unseen / new data
- Model makes accurate, balanced predictions for all availability classes
  - ○ Most of the predicted availabilities are equal to the true availabilities

		True Availability			
n = 37,101		0	1	2	3
Predicted Availability	0	7,329	633	49	28
	1	670	7,403	878	136
	2	49	852	6,060	795
	3	40	145	906	11,128

Confusion matrix that shows a summary of the prediction results

INTERPRETATION OF FINAL MODEL

- Final model to predict bike availability includes the following most important features
  - Latitude
  - Longitude
  - Holiday
  - Hour
  - Month
  - Precipitation



Predicted Avaliablity for Selected Stations at 2019-06-01 09:00:00

True Avaliablity for Selected Stations at 2019-06-01 09:00:00

Predicted availability and true availability are nearly identical - our model was able to predict that there is lower availability in the city and higher in the suburbs in the morning.

ACTIONABLE RECOMMENDATIONS: HOW TO USE THE MODEL

Input

- Determine station to predict availability for
- Determine month, day, and hour to predict availability for
- Obtain relevant weather information for the time of interest
- Construct all features the model takes in from this data

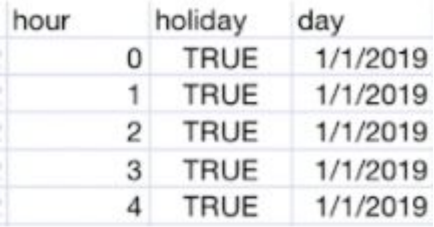
Model

- Feed input into model
- Model predicts availability for the station at the given time

Output

- Model returns a predicted availability for the station at the hour specified
- Predictions are given as classes from 0-25%, 25-50%, etc.

…



hour	holiday	day
0	TRUE	1/1/2019
1	TRUE	1/1/2019
2	TRUE	1/1/2019
3	TRUE	1/1/2019
4	TRUE	1/1/2019

→ dmlc XGBoost →

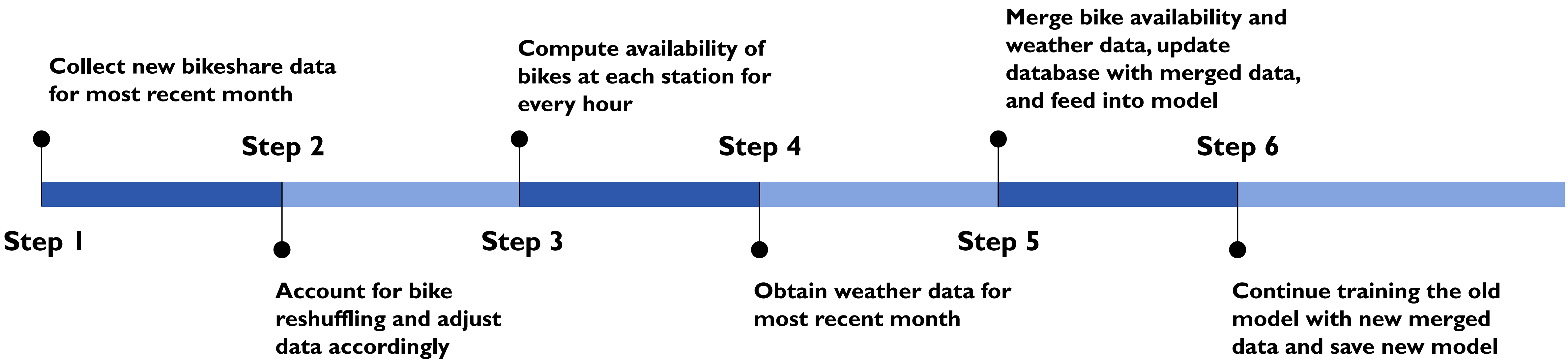
Availability

- ● 0
- ● 1
- ● 2
- ● 3

ACTIONABLE RECOMMENDATIONS: OUR STRATEGY TO LOWER COSTS AND MAXIMIZE REVENUE

- Our action plan
  - Predict hourly availability for all stations over the next two weeks
  - Depending on the patterns of the predicted availability over next two weeks, determine whether to do reshuffle bikes from one station to another
- If the bikes can be reshuffled according to the predicted availability trends, then there will be a reduction in operational costs and maximized revenue
  - Goal: targeted and purposeful reshuffling of bikes rather than random reshuffling
  - Lower operational costs because contractors are reshuffling bikes when necessary
  - Maximized revenue because more customers are able to ride bikes when they need to

PLAN TO MAINTAIN OUR MODEL



Step 1

Collect new bikeshare data for most recent month

Step 2

Account for bike reshuffling and adjust data accordingly

Step 3

Compute availability of bikes at each station for every hour

Step 4

Obtain weather data for most recent month

Step 5

Merge bike availability and weather data, update database with merged data, and feed into model

Step 6

Continue training the old model with new merged data and save new model

NOTE: ENSURE THAT CAPITAL BIKESHARE IS ABLE TO STORE ALL THE DATA IN A DATABASE, SUCH AS MYSQL FOR EASIER/CONVENIENT ACCESS

LIMITATIONS

- We only used 60 of the 654 stations in DC
- We only used 6 months of data, and likely left out a season
- Our model was not based on time-series, so it can likely be outperformed by other forecasting methods
- The weather data was obtained by day, which did not match our availability data, which was by hour

NEXT STEPS

- Time series forecasting instead of extreme gradient boost algorithm
  - Being able to predict bike availability at a particular station at a certain time using consecutive data from previous hours would greatly simplify the model
- Reattempt modeling with larger number of stations
  - Expanding our study to all 654 stations should be the next goal so the model can be generalizable to all of DC

THANK YOU!

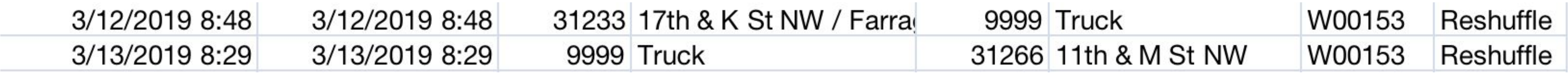
ANY QUESTIONS?

TECHNICAL APPENDIX

STATISTICAL DETAILS REGARDING THE PROCESS IN WHICH WE BUILT OUR PREDICTIVE MODEL

DATA PREPROCESSING: RESHUFFLED DATA

- We add rows to account for bike reshuffling and used trip information to compute bike availability for every station at every hour
- The specific way of doing this is to check the start_station and end_station of the two adjunct rows of the same bike
- If the end_station of the previous row does not match the start_station of the most recent row, we add two reshuffled rows



3/12/2019 8:48	3/12/2019 8:48	31233	17th & K St NW / Farra	9999	Truck	W00153	Reshuffle
3/13/2019 8:29	3/13/2019 8:29	9999	Truck	31266	11th & M St NW	W00153	Reshuffle

NOTE: REFER TO RESHUFFLE.PY FOR CODE

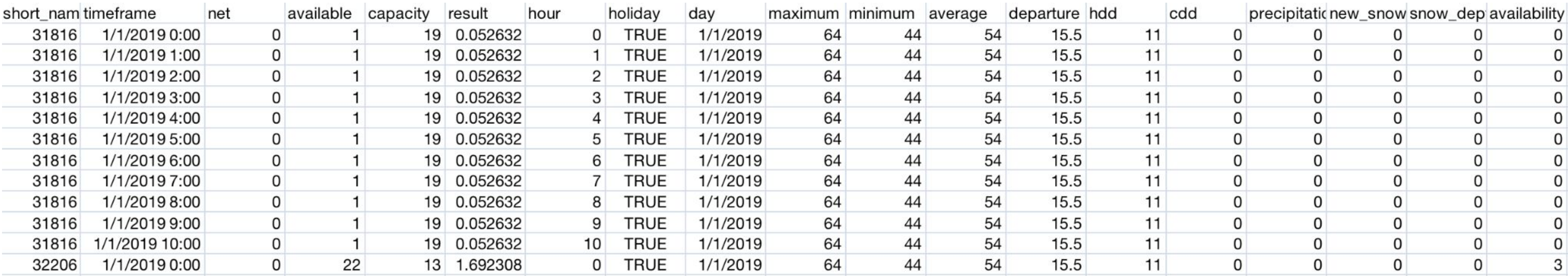
DATA PREPROCESSING: COMPUTING AVAILABILITY

After adding reshuffled rows, we then compute availability for every station at every hour with the following algorithm:

1. Obtain trip data for target month and previous month
2. For previous month, calculate the number of bikes at every station at the end of the month
   a. Group by bike, sort by trip end date, and create a mapping from bike number to last station it arrived at
   b. Count the number of bikes at each station using the mapping from bullet point 1a
3. For current month, calculate the number of bikes that enter and leave every station at every hour
   a. Group by station and time, compute number of bikes leaving the station (i.e. number of trips that start at the station) and the number of bikes that arrive at the station (i.e. number of trips that end at the station)
   b. Compute the difference between bikes entering and exiting the station to compute the net availability change at every station per hour
4. For every station, use the quantity found in bullet point 2 as a starting point for available bikes at the start of the target month, and use the net availability change from bullet point 3 to compute availability at every hour
   a. For a given station at a given time: the availability = the starting availability + the cumulative sum of the net availability change until the target time
5. Correct the data to ensure availability never dips below 0
   a. For every station, compute the minimum availability over the target month
   b. If this quantity is negative, then add its absolute value to every row for that station to account for bikes that did not travel in the previous month
6. Divide availability by capacity, and bin into quartiles

DATA PREPROCESSING: SUMMARIZE THE RESHUFFLED DATA

- After adding the reshuffled rows into the raw dataset, we summarize the data by calculating the number of number of bikes at each station for every hours
- We obtain information such as bike capacity for each station, weather and etc. from the sources below
- We add all the variables we need to the dataset using MySql and calculate the availability for each station for every hour
- Below, we show some rows of our final dataset



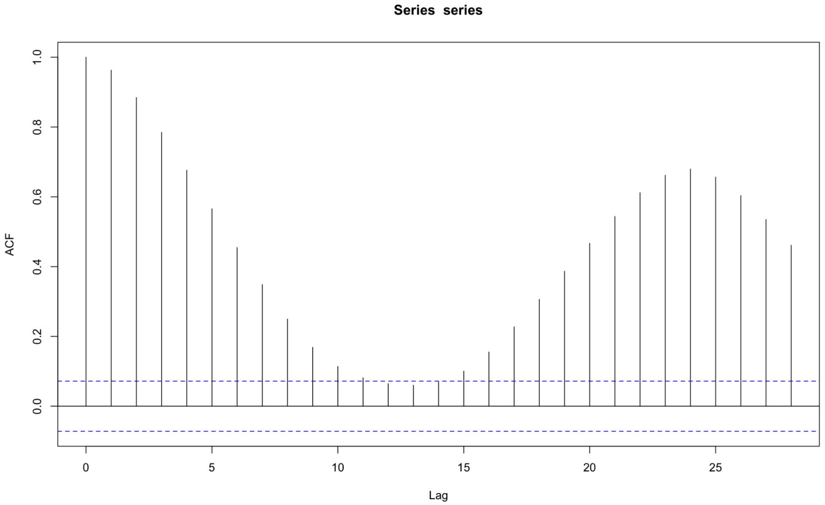
short_nam	timeframe	net	available	capacity	result	hour	holiday	day	maximum	minimum	average	departure	hdd	cdd	precipitatic	new_snow	snow_dep	availability
31816	1/1/2019 0:00	0	1	19	0.052632	0	TRUE	1/1/2019	64	44	54	15.5	11	0	0	0	0	0
31816	1/1/2019 1:00	0	1	19	0.052632	1	TRUE	1/1/2019	64	44	54	15.5	11	0	0	0	0	0
31816	1/1/2019 2:00	0	1	19	0.052632	2	TRUE	1/1/2019	64	44	54	15.5	11	0	0	0	0	0
31816	1/1/2019 3:00	0	1	19	0.052632	3	TRUE	1/1/2019	64	44	54	15.5	11	0	0	0	0	0
31816	1/1/2019 4:00	0	1	19	0.052632	4	TRUE	1/1/2019	64	44	54	15.5	11	0	0	0	0	0
31816	1/1/2019 5:00	0	1	19	0.052632	5	TRUE	1/1/2019	64	44	54	15.5	11	0	0	0	0	0
31816	1/1/2019 6:00	0	1	19	0.052632	6	TRUE	1/1/2019	64	44	54	15.5	11	0	0	0	0	0
31816	1/1/2019 7:00	0	1	19	0.052632	7	TRUE	1/1/2019	64	44	54	15.5	11	0	0	0	0	0
31816	1/1/2019 8:00	0	1	19	0.052632	8	TRUE	1/1/2019	64	44	54	15.5	11	0	0	0	0	0
31816	1/1/2019 9:00	0	1	19	0.052632	9	TRUE	1/1/2019	64	44	54	15.5	11	0	0	0	0	0
31816	1/1/2019 10:00	0	1	19	0.052632	10	TRUE	1/1/2019	64	44	54	15.5	11	0	0	0	0	0
32206	1/1/2019 0:00	0	22	13	1.692308	0	TRUE	1/1/2019	64	44	54	15.5	11	0	0	0	0	3

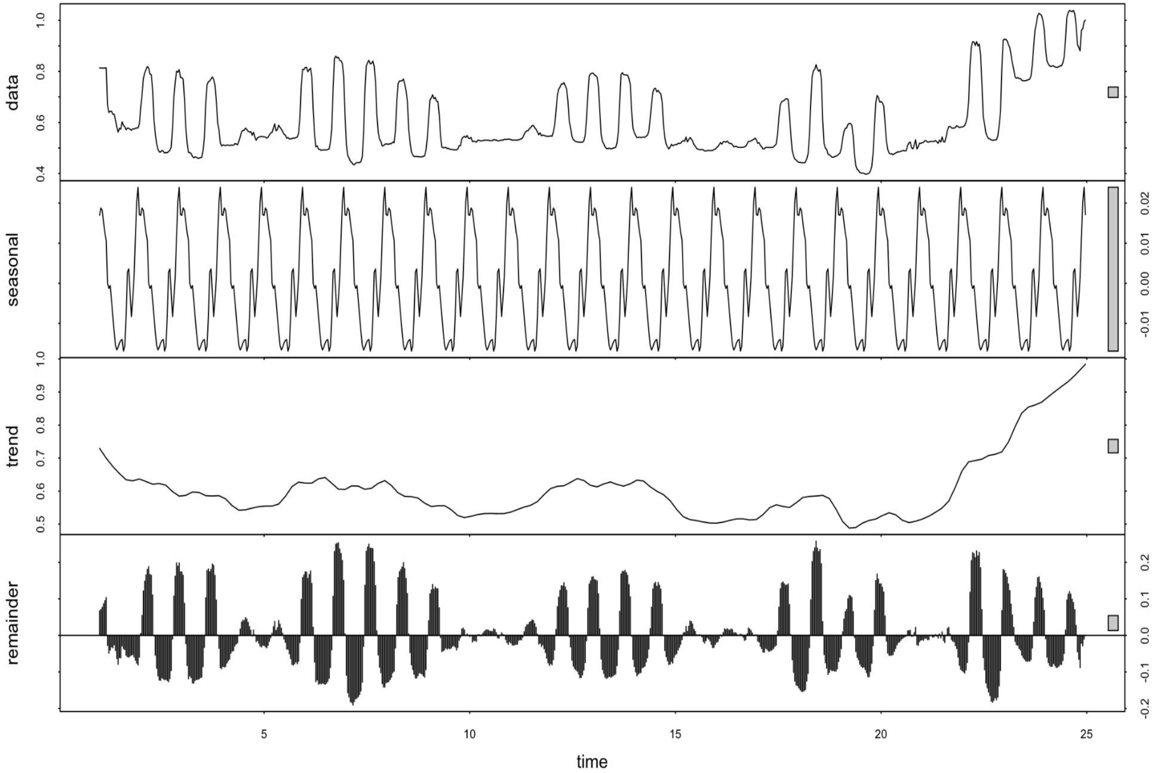
SOURCES: NATIONAL OCEANIC AND ATMOSPHERIC ADMINISTRATION, PYTHON PACKAGES DATETIME AND HOLIDAYS, CAPITAL BIKESHARE'S GITHUB

NOTE: REFER TO FINALDATA.SQL FOR THE CODE

ARMA Model

- We attempted to do simple time-series models as our first attempt to modeling the data
- Using only univariate data (Average capacity per day) for ARMA model, the data was very highly correlated to itself, so we decide it was not a good idea to use an AR or ARMA model

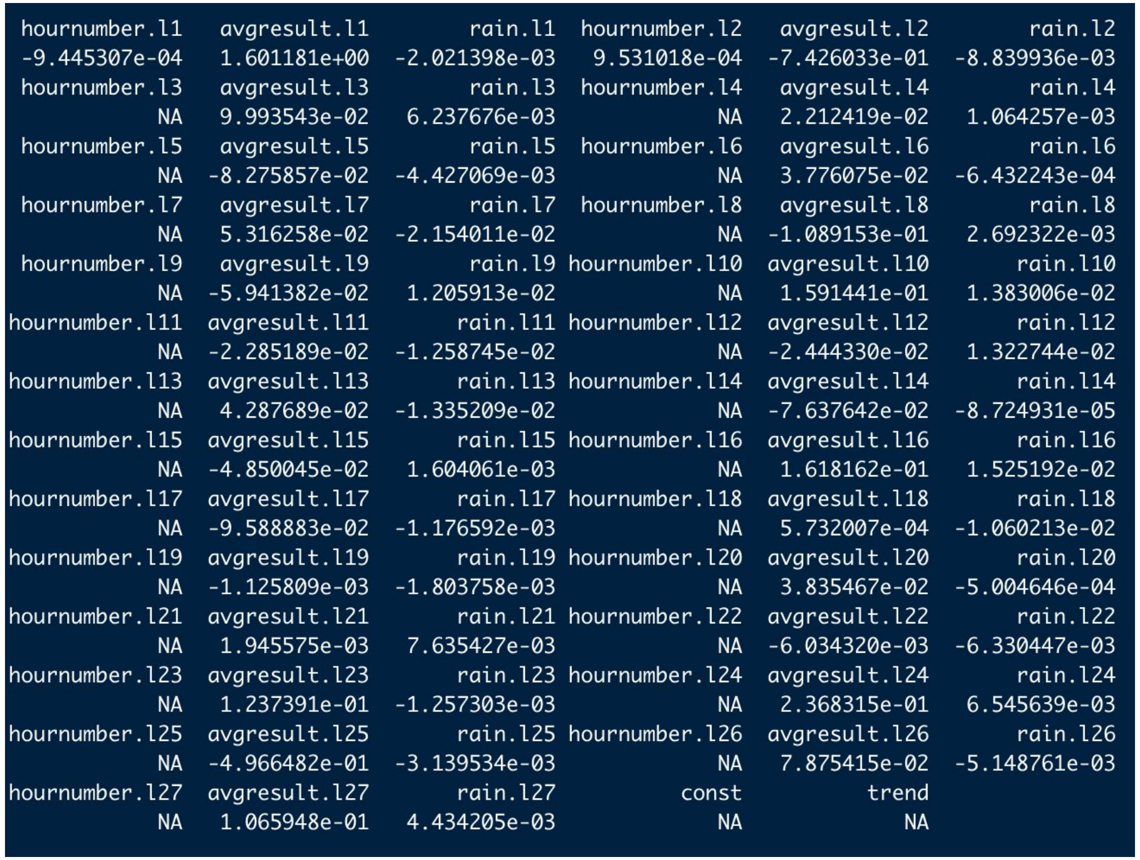




VAR Model

- We switched to VAR to account for more of the variables mentioned earlier in the presentation, but it resulted in many terms with no coefficients for an intercept or trend

- The output on the right shows the VAR(27) model for predicting Average Capacity



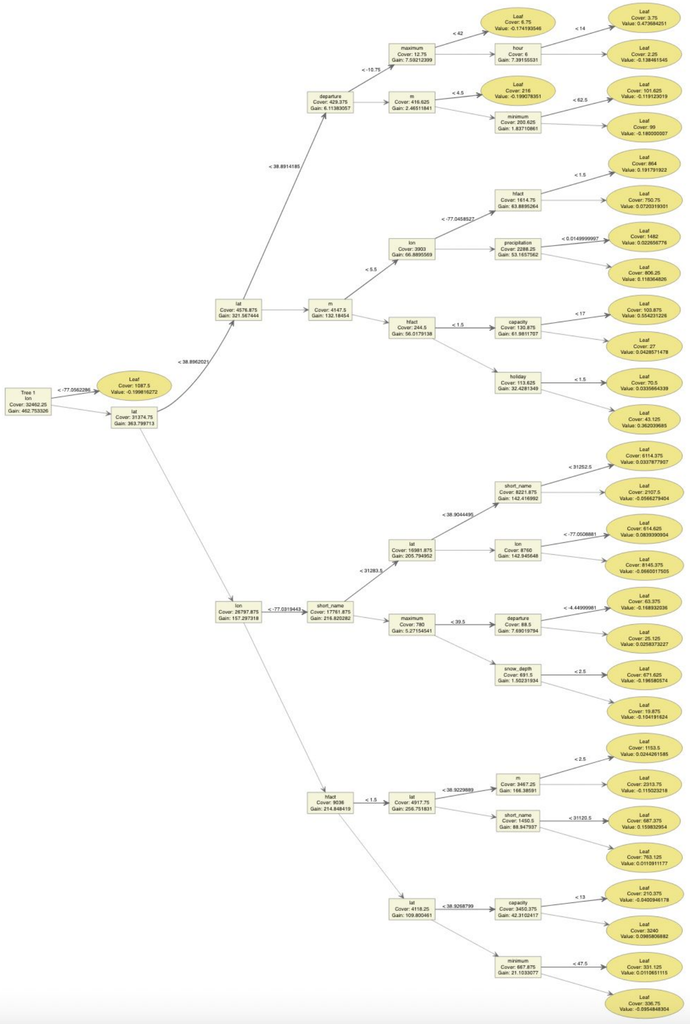
```
 hournumber.l1   avgresult.l1        rain.l1  hournumber.l2   avgresult.l2        rain.l2
 -9.445307e-04   1.601181e+00  -2.021398e-03   9.531018e-04  -7.426033e-01  -8.839936e-03
 hournumber.l3   avgresult.l3        rain.l3  hournumber.l4   avgresult.l4        rain.l4
            NA   9.993543e-02   6.237676e-03             NA   2.212419e-02   1.064257e-03
 hournumber.l5   avgresult.l5        rain.l5  hournumber.l6   avgresult.l6        rain.l6
            NA  -8.275857e-02  -4.427069e-03             NA   3.776075e-02  -6.432243e-04
 hournumber.l7   avgresult.l7        rain.l7  hournumber.l8   avgresult.l8        rain.l8
            NA   5.316258e-02  -2.154011e-02             NA  -1.089153e-01   2.692322e-03
 hournumber.l9   avgresult.l9        rain.l9 hournumber.l10  avgresult.l10       rain.l10
            NA  -5.941382e-02   1.205913e-02             NA   1.591441e-01   1.383006e-02
hournumber.l11  avgresult.l11       rain.l11 hournumber.l12  avgresult.l12       rain.l12
            NA  -2.285189e-02  -1.258745e-02             NA  -2.444330e-02   1.322744e-02
hournumber.l13  avgresult.l13       rain.l13 hournumber.l14  avgresult.l14       rain.l14
            NA   4.287689e-02  -1.335209e-02             NA  -7.637642e-02  -8.724931e-05
hournumber.l15  avgresult.l15       rain.l15 hournumber.l16  avgresult.l16       rain.l16
            NA  -4.850045e-02   1.604061e-03             NA   1.618162e-01   1.525192e-02
hournumber.l17  avgresult.l17       rain.l17 hournumber.l18  avgresult.l18       rain.l18
            NA  -9.588883e-02  -1.176592e-03             NA   5.732007e-04  -1.060213e-02
hournumber.l19  avgresult.l19       rain.l19 hournumber.l20  avgresult.l20       rain.l20
            NA  -1.125809e-03  -1.803758e-03             NA   3.835467e-02  -5.004646e-04
hournumber.l21  avgresult.l21       rain.l21 hournumber.l22  avgresult.l22       rain.l22
            NA   1.945575e-03   7.635427e-03             NA  -6.034320e-03  -6.330447e-03
hournumber.l23  avgresult.l23       rain.l23 hournumber.l24  avgresult.l24       rain.l24
            NA   1.237391e-01  -1.257303e-03             NA   2.368315e-01   6.545639e-03
hournumber.l25  avgresult.l25       rain.l25 hournumber.l26  avgresult.l26       rain.l26
            NA  -4.966482e-01  -3.139534e-03             NA   7.875415e-02  -5.148761e-03
hournumber.l27  avgresult.l27       rain.l27          const          trend
            NA   1.065948e-01   4.434205e-03             NA             NA
```

Logistic Regression

- We tried Logistic Regression to solve the multiclass (i.e. 4 categories) classification problem, but after fitting the model, we find that the performance was weak, with accuracy of only about 0.62
- The weak performance makes sense, since the Logistic Regression fits a linear decision boundary
  - Our data violated many assumptions of a linear model
  - A linear decision boundary is not suitable for our problem

XGBoost

- Generally, XGBoost is a kind of GBDT (Gradient Boosting Decision Trees) algorithm
- XGBoost uses multiple decision trees to fit the model; it uses new trees to fit the residuals from the previous trees
- In our project, we used XGBoost to solve the muilticlass classification problem
- The model was trained for 500 iterations using a maximum tree depth of 8 and a multiclass softmax loss function
- The tree on the right is a part of our final model
    - For more details, please refer to the footnote



Tree 1 lon Cover: 32462.25 Gain: 462.753326
< -77.0562286
Leaf Cover: 1087.5 Value: -0.199816272
lat Cover: 31374.75 Gain: 363.799713
< 38.8962021
lat Cover: 4576.875 Gain: 321.567444
< 38.8914185
departure Cover: 429.375 Gain: 6.11383057
< -10.75
maximum Cover: 12.75 Gain: 7.59212399
< 42
Leaf Cover: 6.75 Value: -0.174193546
hour Cover: 6 Gain: 7.39155531
< 14
Leaf Cover: 3.75 Value: 0.473684251
Leaf Cover: 2.25 Value: -0.138461545
m Cover: 416.625 Gain: 2.46511841
< 4.5
Leaf Cover: 216 Value: -0.199078351
minimum Cover: 200.625 Gain: 1.83710861
< 62.5
Leaf Cover: 101.625 Value: -0.119123019
Leaf Cover: 99 Value: -0.180000007
m Cover: 4147.5 Gain: 132.18454
< 5.5
lon Cover: 3903 Gain: 66.8895569
< -77.0458527
hfact Cover: 1614.75 Gain: 63.8895264
< 1.5
Leaf Cover: 864 Value: 0.191791922
Leaf Cover: 750.75 Value: 0.0720319301
precipitation Cover: 2288.25 Gain: 53.1657562
< 0.0149999997
Leaf Cover: 1482 Value: 0.022656776
Leaf Cover: 806.25 Value: 0.118364826
hfact Cover: 244.5 Gain: 56.0179138
< 1.5
capacity Cover: 130.875 Gain: 61.9811707
< 17
Leaf Cover: 103.875 Value: 0.554231226
Leaf Cover: 27 Value: 0.0428571478
holiday Cover: 113.625 Gain: 32.4281349
< 1.5
Leaf Cover: 70.5 Value: 0.0335664339
Leaf Cover: 43.125 Value: 0.362039685
lon Cover: 26797.875 Gain: 157.297318
< -77.0319443
short_name Cover: 17761.875 Gain: 216.820282
< 31283.5
lat Cover: 16981.875 Gain: 205.794952
< 38.9044495
short_name Cover: 8221.875 Gain: 142.416992
< 31252.5
Leaf Cover: 6114.375 Value: 0.0337877907
Leaf Cover: 2107.5 Value: -0.0566279404
lon Cover: 8760 Gain: 142.945648
< -77.0508881
Leaf Cover: 614.625 Value: 0.0839390904
Leaf Cover: 8145.375 Value: -0.0660017505
maximum Cover: 780 Gain: 5.27154541
< 39.5
departure Cover: 88.5 Gain: 7.69019794
< -4.44999981
Leaf Cover: 63.375 Value: -0.168932036
Leaf Cover: 25.125 Value: 0.0258373227
snow_depth Cover: 691.5 Gain: 1.50231934
< 2.5
Leaf Cover: 671.625 Value: -0.196580574
Leaf Cover: 19.875 Value: -0.104191624
hfact Cover: 9036 Gain: 214.848419
< 1.5
lat Cover: 4917.75 Gain: 256.751831
< 38.9229889
m Cover: 3467.25 Gain: 166.38591
< 2.5
Leaf Cover: 1153.5 Value: 0.0244261585
Leaf Cover: 2313.75 Value: -0.115023218
short_name Cover: 1450.5 Gain: 88.947937
< 31120.5
Leaf Cover: 687.375 Value: 0.159832954
Leaf Cover: 763.125 Value: 0.0110911177
lat Cover: 4118.25 Gain: 109.800461
< 38.9268799
capacity Cover: 3450.375 Gain: 42.3102417
< 13
Leaf Cover: 210.375 Value: -0.0400946178
Leaf Cover: 3240 Value: 0.0985805882
minimum Cover: 667.875 Gain: 21.1033077
< 47.5
Leaf Cover: 331.125 Value: 0.0110651115
Leaf Cover: 336.75 Value: -0.0954848304

NOTE: REFER TO XGBIKE.R FOR MODEL AND CODE

The Availability Maps

- The availability plots on page 8 was plotted using ggplot and the latitude and longitude information for each station
- The map of Washington DC was obtained from the Google Maps API
- By using our dataset and the code mentioned in the note, we could plot the availability of each station at every hour from 2019-01 to 2019-07

NOTE: REFER TO XGBIKE_MAPS.R FOR PLOTS AND CODE