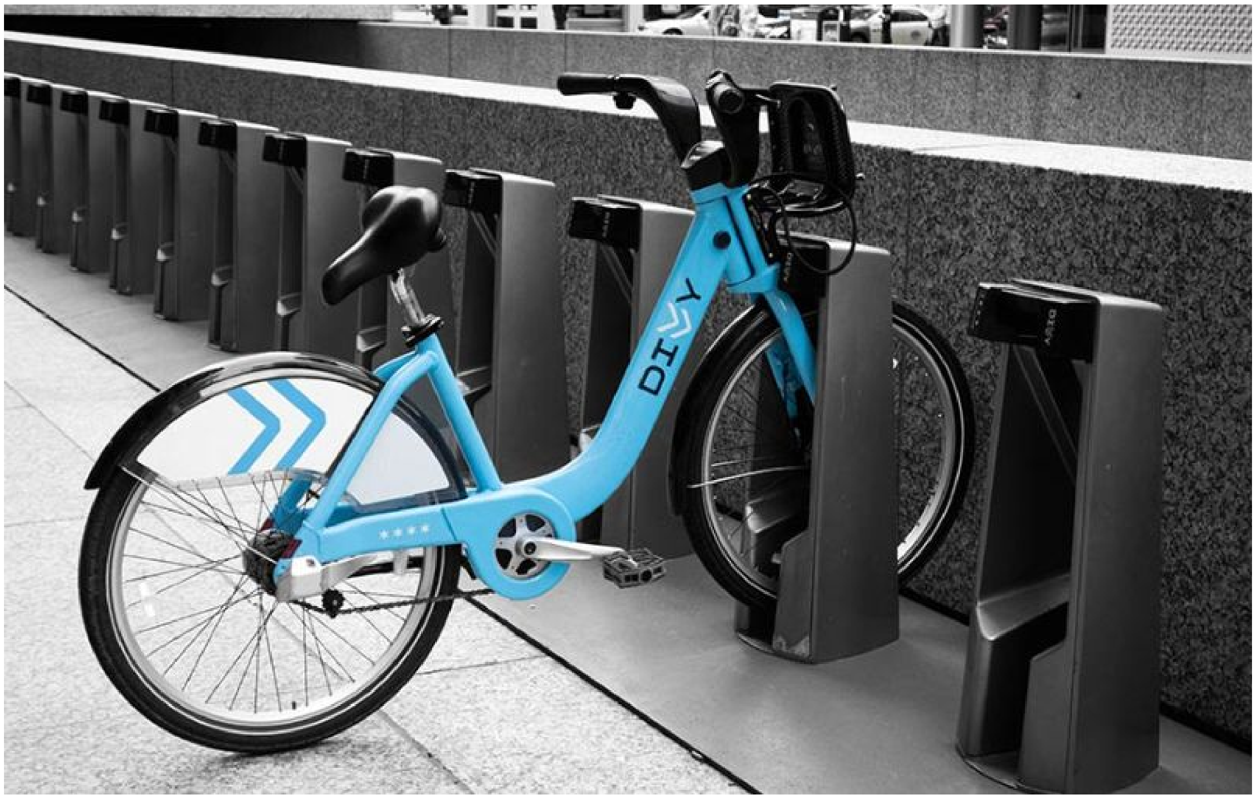
Divvy Bike Capstone Project

Tina Carr

November 2018



Introduction to Divvy and project aim

Divvy is Chicago's bike sharing system which is owned by the Chicago Department of Transportation with the aim to "promote economic recovery, reduce traffic congestion and improve air quality". With over 6,000 bikes and 580 docking stations in the Chicagoland / Evanston area that are available to the public 24/7 the higher the accuracy rate Divvy's forecasting models can achieve translates into more happy customers looking for a sustainable and fun way to get around the city. Therefore, the aim of this capstone project is to develop a forecasting model for one of Divvy's most popular bike stations located at Lake Shore Dr. & Monroe for the purposes of inventory management and improving the customer experience by ensuring there are always enough bikes available at this station.

The Data

This data set has a total of 13,821,994 observations and 22 columns including:

Trip ID	Community areas	To Longitude	From Latitude	From Longitude	From Location
From Station ID	Start Time	Wards	To Location	Boundaries	Zip Codes
Gender	From Station Name	Stop Time	Bike ID	Trip Duration	To Latitude
Birth Year	To Station ID	To Station Name	User Type		

Data collection began when Divvy first started its operations in June 2013, this dataset contains records until December 2017.

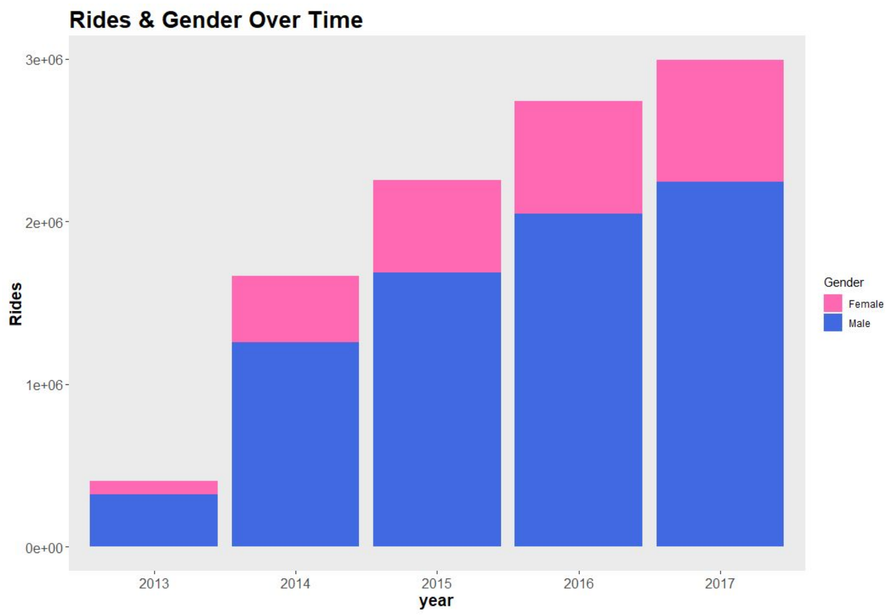
For the purpose of my time series I made 2 subsets of the data. The first subset being the **bikes taken to (203,014 observations)** Lake Shore Dr & Monroe and the 2nd bieng **bikes taken from (210,255 observations)** the Lake Shore Dr. and Monroe station. For the purposes of my forecasting model the "stop.time" column is important for bikes taken to the station since that data captures the moment that the inventory (bike) was returned to the station and the "start.time" column is important for bikes taken to the station since that data captures the moment that the inventory (bike) was taken from the station

Exploratory Data Analysis

In addition to the time series modelling in this capstone I also wanted to take a deeper dive into the overall data set to refine some insights on the Divvy brand and customer. In order to do this I concentrated on the Gender, Birth Year and Trip Duration columns of the data set.

Gender

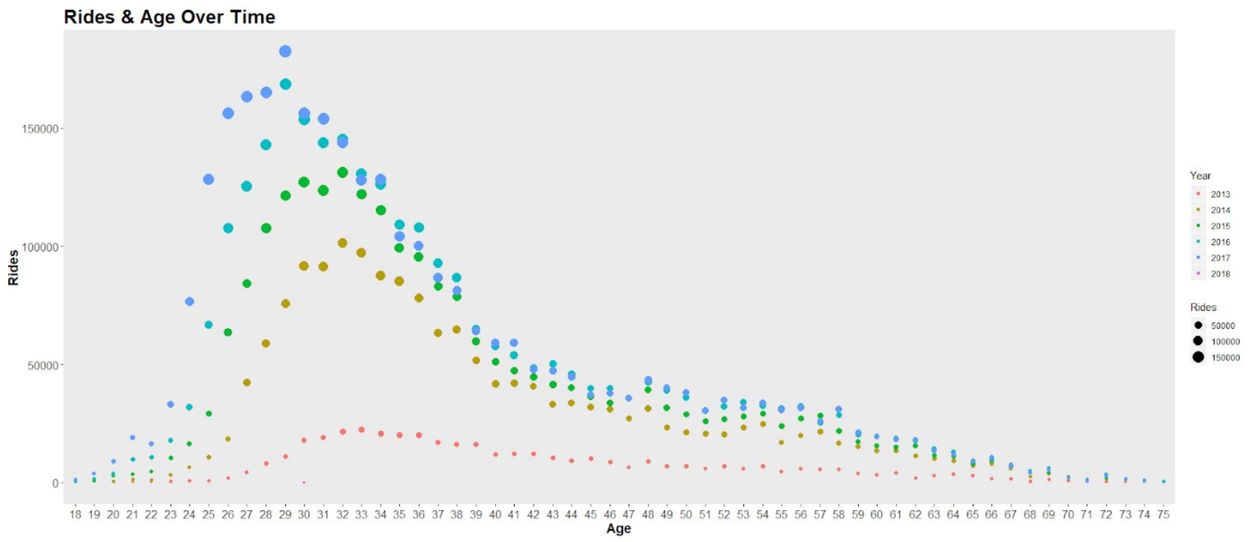
The overall male/female ratio of Divvy trips was 7,552,990 / 2,495,777 as depicted in the graph on the following page we can see that although Divvy has significantly increased its female riders since 2013 from 21% to 25% in 2017 this gender ratio is not reflective of the overall Chicago gender ratio of Chicago which cites the Chicago population as 51% female as reported in US census data.



Male / Female Trip Ratio:
7,552,990 / 2,495,777

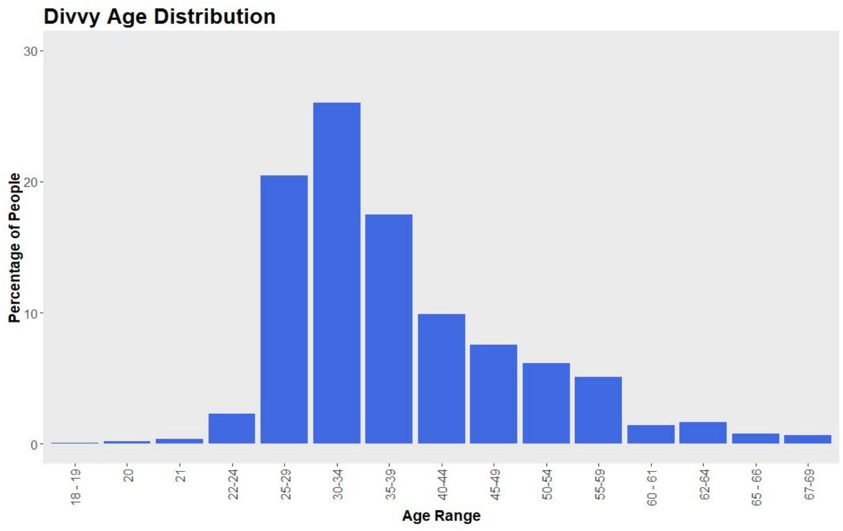
increase in female riders trips from only 21% in 2013 to 25% in 2017

Age

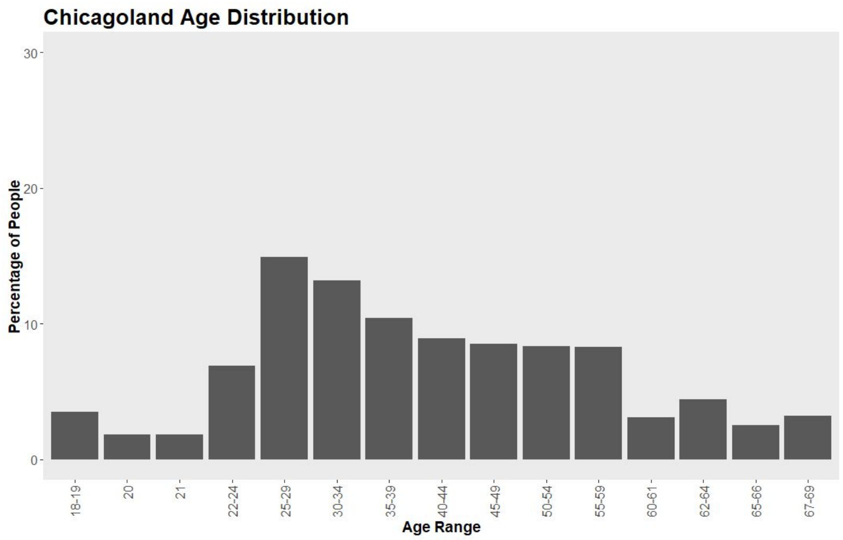


In the graph above the age and number of riders on the x and y axis. Each year Divvy was in operation is indicated by the different colors of the dots and the larger the dot appears on the graph represents a higher number of trips taken from that age segment.

From this visualization initially it seems that that over time the "peak" age for riders has shifted from 33 in 2013 to around 28 in 2018 and Divvy seems to capture the late 20s / early 30's demographic quite well. How well to Divvy's age segments reflect the overall population of Chicago?



In order to get a better idea of what a "baseline" customer age distribution would look like the visualizations the the left compare the Divvy age segmentation to the overall City of Chicago age segmentation (source: U.S. Census). The y-axis is listed as an overall percentage of the Divvy / Chicago population in order to represent these data sets in as much of an unbiased manner as possible.



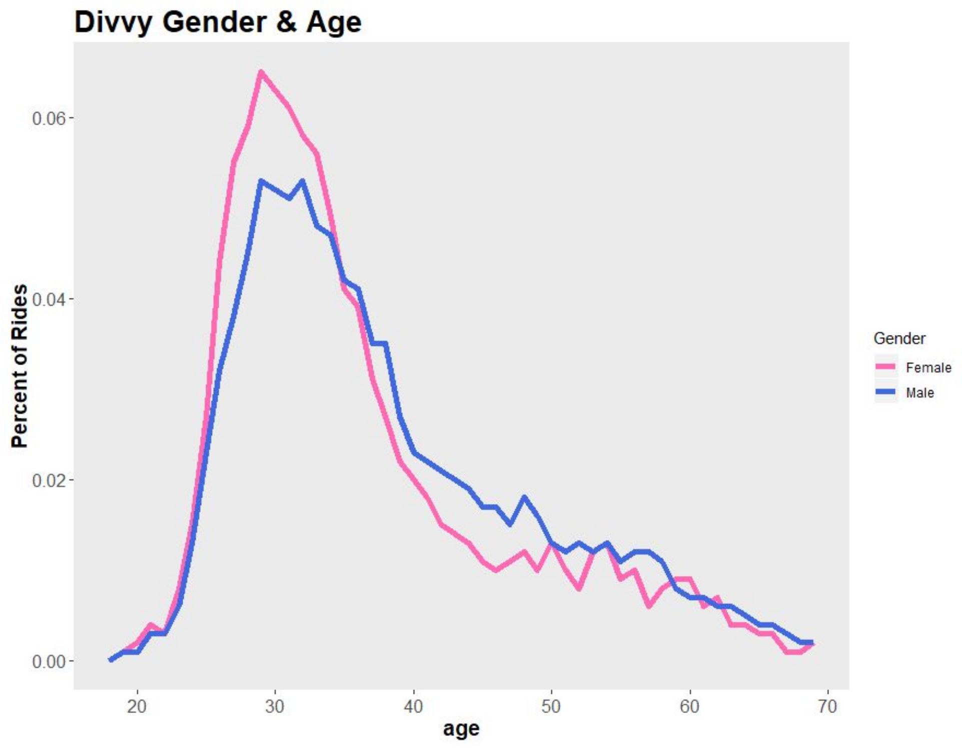
The Divvy age distribution is skewed to the left starting in the early 20's compared to the overall population in Chicago which demonstrates there is untapped potential in the mid-age market segment as well as the 18-24 segment.

Divvy age standard deviation is 10.8 compared to the Chicago age standard deviation of 14.2.

Taking Divvy's product / service into consideration the wider spread / standard deviation of the Chicago standard deviation could in part be explained by the fact older people, which are considered part of the Chicago population, are not interested or physically unable to bike ride thus preventing Divvy bike from capturing this segment. In any case taking standard deviation into consideration if Divvy targets new age segments for which their customer base is currently much lower compared to the Chicago population and can safely assume is physically able to ride a bike (18-24 segment and the 45+ segments for example) the Divvy standard deviation will naturally increase towards the Chicago standard deviation.

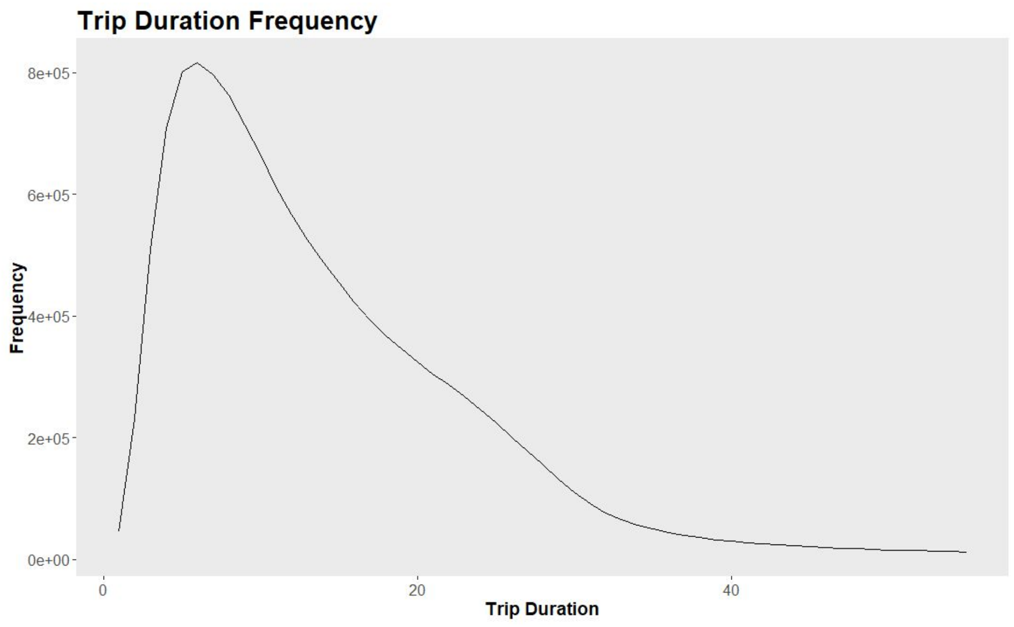
Age and Gender

In order to see how the components age and gender interact I created a visualization comparing the percentage of rides for male and female riders and their corresponding age segments on the below. The male and female data experiences similar peaks and dips with some exceptions. A peak in male an female riders happens around the late 20's - however males experience a second peak in the early 30's. There is also a peak in male riders in their late 40's while female riders experience a dip in that age - a similar instance happens in late 50's. Overall it seems that Divvy male riders come from a more diverse age segment pool compared to females which peak in their late 20s/early 30s and decrease significantly after that. If Divvy is able to attract more female riders in the future it will be interesting to see if the female segment develops a wider spread as well and is not so dependant on the late 20s/early 30s female demographic.



Trip duration

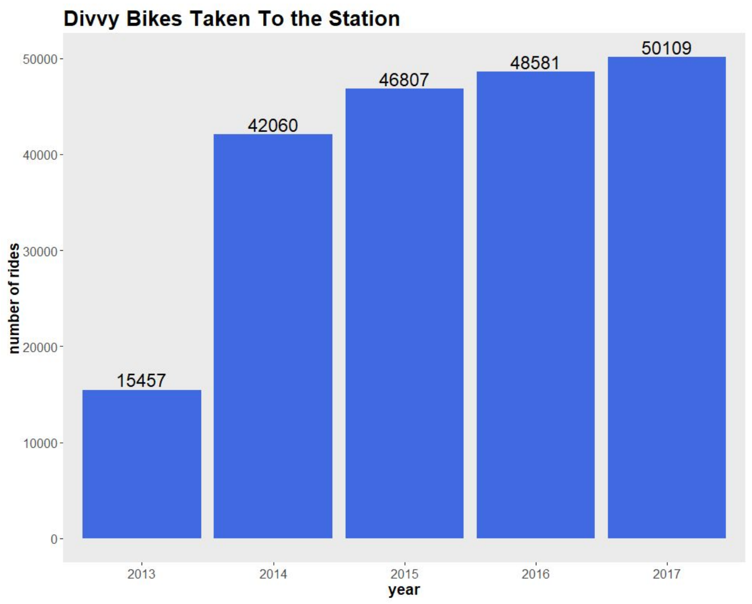
Trip duration below is recorded in minutes below.

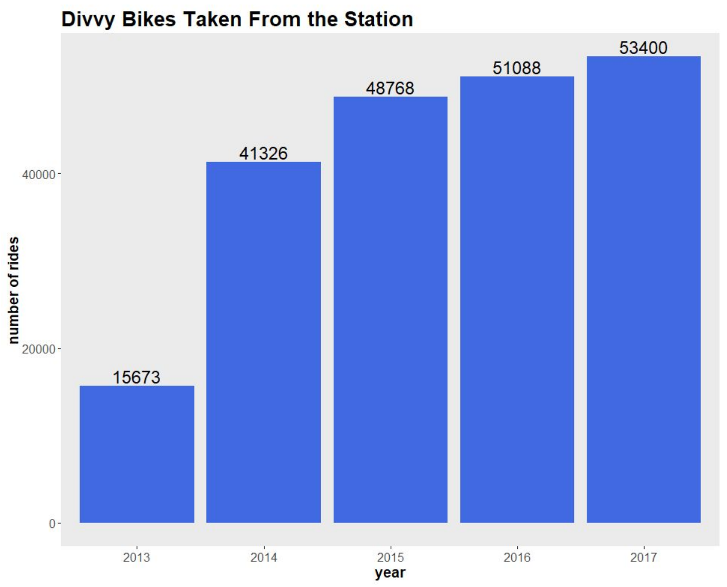


As shown in the graph above there is a clear peak between 0-20 followed by a steep drop. This trip length visualization makes sense when taking Divvy's price structure into consideration, keeping in mind that this data set's last observation was in 2017. Divvy was receiving complaints about their 30 minute ride pricing-structure and in 2018 they decided to change their pricing structure in order to encourage rides longer than 30 minutes. Since the price structure which divvy had set up for the daily and annual offering encouraged rides of 30 minutes or less prior to 2018, naturally that was also reflected in the trip duration visualization since customers wanted to avoid any extra cost for exceeding the standard trip duration.

It will be especially interesting to follow overall trip duration on Divvy rides after the announcement of the explorer pass to see if this pass which encourages riders interested in longer trips to use Divvy, has an effect on the overall trip duration of Divvy's customer base.

TO and FROM Datasets for the Station





The TO and FROM datasets for the Lake Shore Dr. & Monroe Divvy station are visualized above by observations per year. In both data sets the significant increase from 2013 - 2014 can in part be explained by the fact the 2013 only covered 6 months of the year while 2014 - 2017 data was recorded for a full 12 month calendar year. There is also a positive trend of rides as each year passes which is natural for a brand just starting out as more and more people become aware, considering the fact Divvy started operations in 2013.

TO had a total of 203,014 observations while FROM had a total of 210,255 observations.

Time Series (ts) Modeling

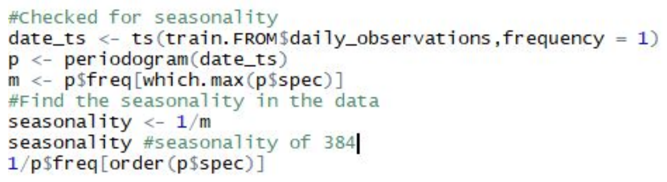
I approached the time series modeling in the following steps:

1. Data Wrangling
    a. In order to properly prep this data for the time series I first used tidyr to organize my data by “date” and “daily_observations”. While examining this data I noticed there were days missing in areas as well as a leap year. I removed the leap year from the data in order to prevent any issues with arima later.
2. Created test & train data sets
    a. Since my data covers approx. 3.5 years I decided to leave my train data set larger and forecasted 4 months out.
3. Ran a periodogram to find the optimal seasonality
4. Created the ts function
5. “Baseline model” with Arima (1,1,1)
6. Auto.Arima model with seasonality
7. Models based on acf & pacf estimations
8. Additional model exploration

FROM Forecasting

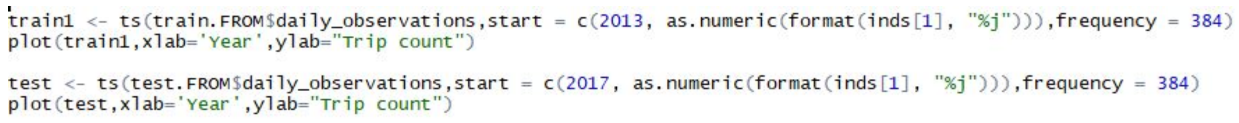
Creating the ts model & frequency

Before running the ts model I used the following code to find the seasonality in R:



```
#Checked for seasonality
date_ts <- ts(train.FROM$daily_observations,frequency = 1)
p <- periodogram(date_ts)
m <- p$freq[which.max(p$spec)]
#Find the seasonality in the data
seasonality <- 1/m
seasonality #seasonality of 384
1/p$freq[order(p$spec)]
```

The result was an optimum seasonality of 384 for which I also accounted for while creating my ts functions below.



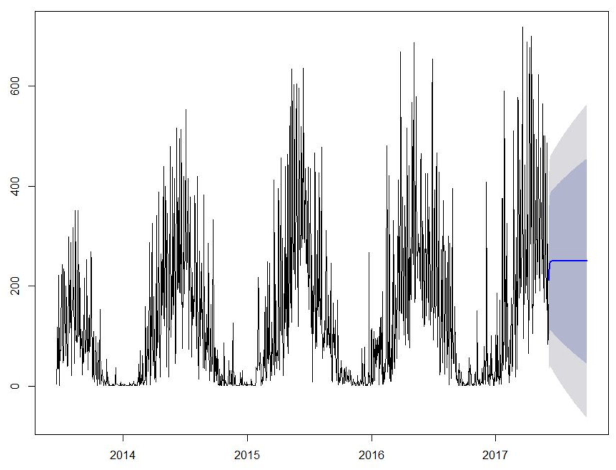
```
train1 <- ts(train.FROM$daily_observations,start = c(2013, as.numeric(format(inds[1], "%j"))),frequency = 384)
plot(train1,xlab='Year',ylab="Trip count")

test <- ts(test.FROM$daily_observations,start = c(2017, as.numeric(format(inds[1], "%j"))),frequency = 384)
plot(test,xlab='Year',ylab="Trip count")
```

Baseline Model

In order to create a baseline model I used the Arima (1,1,1) model as a comparison. Below were the resulting AIC, RMSE values from this model in addition to the forecast.

Model	Train AIC	Train RMSE	Test RMSE	Train - Test RMSE
(1,1,1)	18021.96	89.57	193.25	103.68

Forecasts from ARIMA(1,1,1)



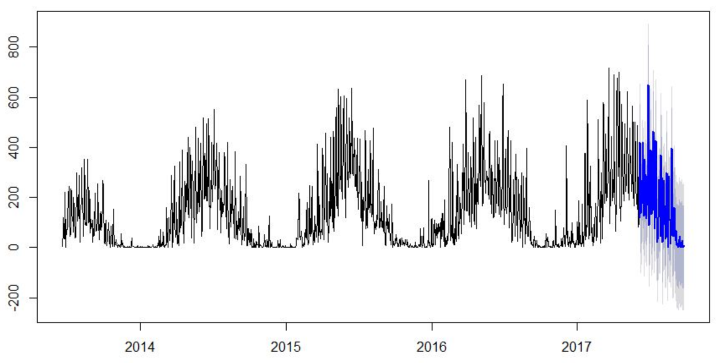
Ideally the baseline model should be improved by adding seasonality, lowering the test RMSE and improving the train AIC.

Auto.Arima Model

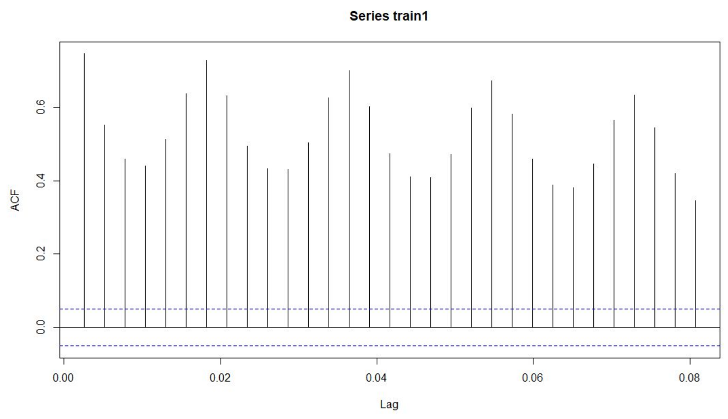
The auto.arima model recommended p,d,q values of 2,0,2 and seasonality of 0,1,0.

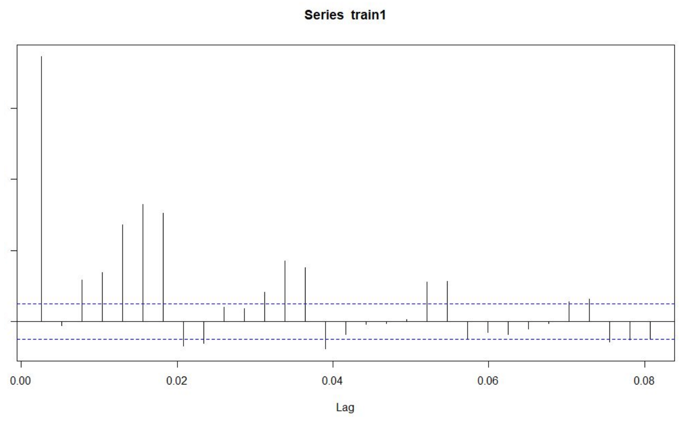
Model	Train AIC	Train RMSE	Test RMSE	Train - Test RMSE
(2,0,2)(0,1,0 p =384)	13973.81	95.59	132.63	37.04

Forecasts from ARIMA(2,0,2)(0,1,0)[384]



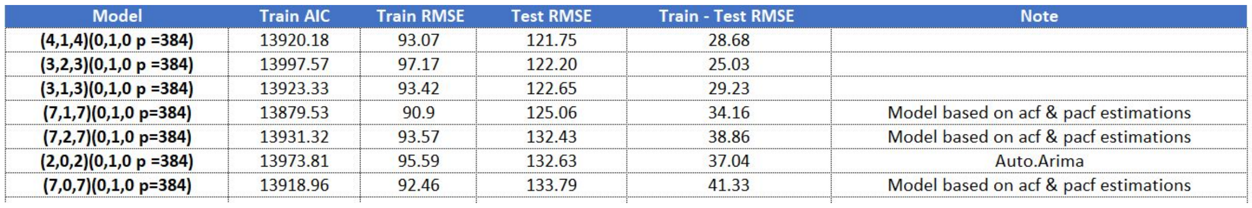
Models based on acf & pacf estimations





Models were also ran which which were based off of the observations for the ACF and PACF graphs below with a p and q value of 7. Later on in the final comparison table models with these estimations will also be included.

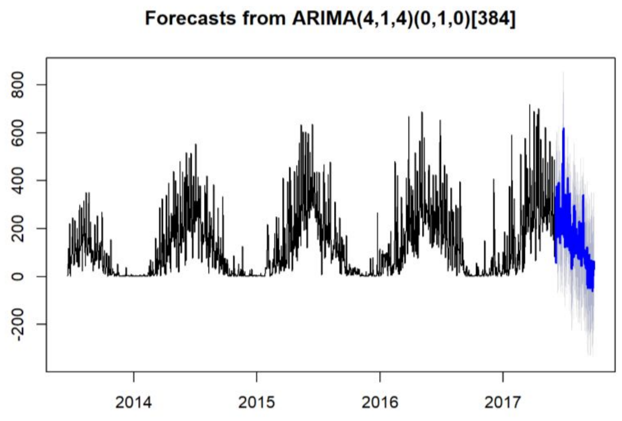
FROM Final Comparison Table



Model	Train AIC	Train RMSE	Test RMSE	Train - Test RMSE	Note
(4,1,4)(0,1,0 p =384)	13920.18	93.07	121.75	28.68	
(3,2,3)(0,1,0 p =384)	13997.57	97.17	122.20	25.03	
(3,1,3)(0,1,0 p =384)	13923.33	93.42	122.65	29.23	
(7,1,7)(0,1,0 p=384)	13879.53	90.9	125.06	34.16	Model based on acf & pacf estimations
(7,2,7)(0,1,0 p=384)	13931.32	93.57	132.43	38.86	Model based on acf & pacf estimations
(2,0,2)(0,1,0 p =384)	13973.81	95.59	132.63	37.04	Auto.Arima
(7,0,7)(0,1,0 p=384)	13918.96	92.46	133.79	41.33	Model based on acf & pacf estimations

The table above summarizes the various models I used in forecasting observations from this Divvy bike station. The various model types are ordered lowest to highest based on the test RMSE. The table listed above only includes the top 3 performing models compared to the models which were mentioned in the previous sections, for a full list of the models which were used please see the appendix.

The models based on the acf and pcf values did perform better than the auto.arima models in most cases but in the end the **(4,1,4)(0,1,0 p=384)** model performed the best when it came to test accuracy and is therefore the best fit model out of the models listed above for the bikes taken from the Divvy station.



TO Forecasting

Before running the ts model I used the following code to find the seasonality in R:

```
#Checked for seasonality
date_tsto <- ts(train.TO$daily_observations,frequency = 1)
p <- periodogram(date_tsto)
m <- p$freq[which.max(p$spec)]
#Find the seasonality in the data
seasonality <- 1/m
seasonality #seasonality of 384
1/p$freq[order(p$spec)]
```

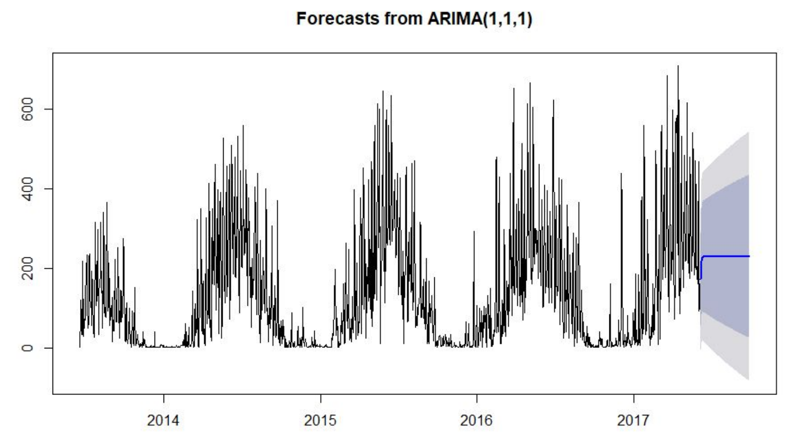
The result was an optimum seasonality of 384 for which I also accounted for while creating my ts functions below.

```
trainto <- ts(train.TO$daily_observations,start = c(2013, as.numeric(format(inds[1], "%j"))),frequency = 384)
plot(train1,xlab='Year',ylab="Trip count")
acf(trainto)
pacf(trainto)

testto <- ts(test.FROM$daily_observations,start = c(2017, as.numeric(format(inds[1], "%j"))),frequency = 384)
plot(testto,xlab='Year',ylab="Trip count")
```

Baseline Model

In order to create a baseline model I used the Arima (1,1,1) model as a comparison. Below were the resulting AIC, RMSE values from this model in addition to the forecast.

Model	Train AIC	Train RMSE	Test RMSE	Train - Test RMSE
(1,1,1)	18044.66	90.24	183.61	93.37

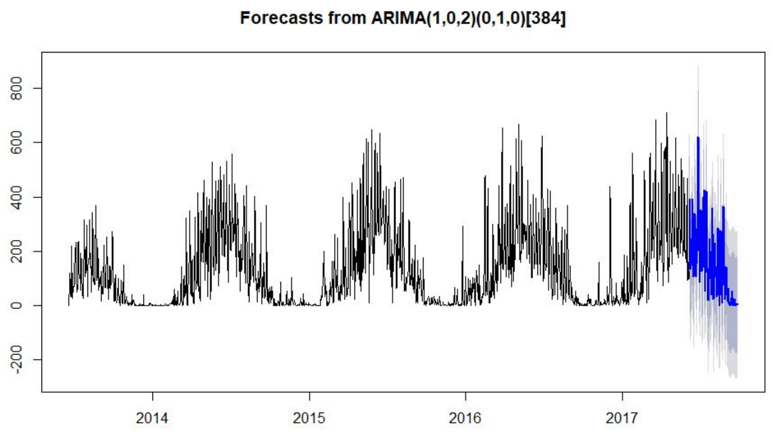


Ideally the baseline model should be improved by adding seasonality, lowering the test RMSE and improving the train AIC.

Auto.Arima Model

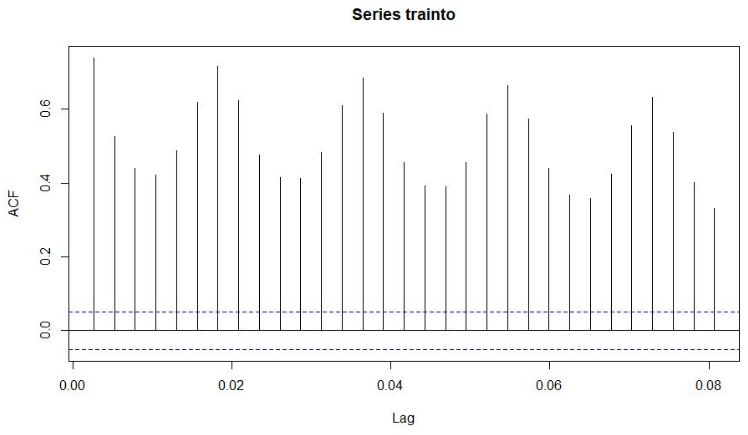
The next step was checking what auto.arima would recommend. Although the FROM and TO data sets appeared quite similar when it came to seasonality the TO dataset recommendation from auto.arima has a slightly lower p value than the FROM data.

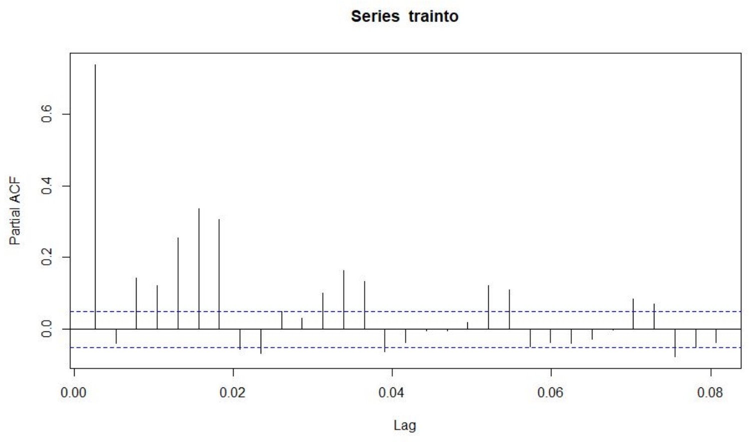
Model	Train AIC	Train RMSE	Test RMSE	Train - Test RMSE
(1,0,2) (0,1,0 p = 384)	14148.26	103.29	130.51	27.22



Models based on acf & pacf estimations

Models were also ran which which were based off of the observations for the ACF and PACF graphs below with a p and q value of 7. Later on in the final comparison table models with these estimations will also be included.



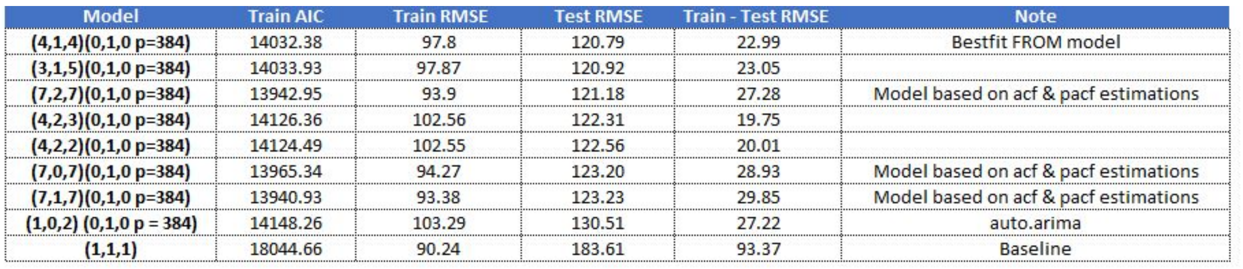


Using the best fit FROM model on TO data

In addition to the baseline and auto.arima model, since the TO and FROM data sets shared a common seasonality I used the best fit from model (4,1,4)(0,1,0 p= 384) . This model has a lower test RMSE and therefore was a better fit model to use compared to auto.arima.

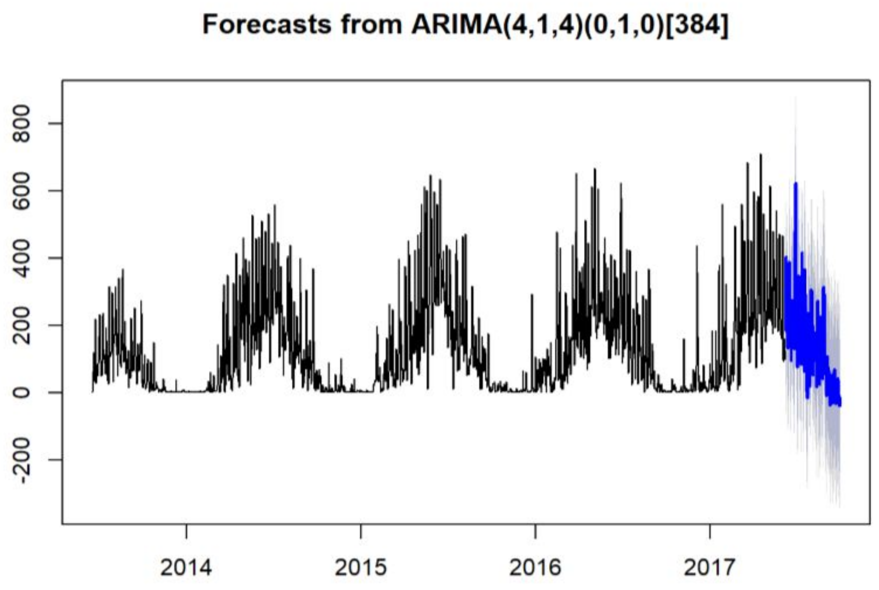
Model	Train AIC	Train RMSE	Test RMSE	Train - Test RMSE	Note
(4,1,4)(0,1,0 p=384)	14032.38	97.8	120.79	22.99	Bestfit FROM model

TO Final Comparison Table



Model	Train AIC	Train RMSE	Test RMSE	Train - Test RMSE	Note
(4,1,4)(0,1,0 p=384)	14032.38	97.8	120.79	22.99	Bestfit FROM model
(3,1,5)(0,1,0 p=384)	14033.93	97.87	120.92	23.05	
(7,2,7)(0,1,0 p=384)	13942.95	93.9	121.18	27.28	Model based on acf & pacf estimations
(4,2,3)(0,1,0 p=384)	14126.36	102.56	122.31	19.75	
(4,2,2)(0,1,0 p=384)	14124.49	102.55	122.56	20.01	
(7,0,7)(0,1,0 p=384)	13965.34	94.27	123.20	28.93	Model based on acf & pacf estimations
(7,1,7)(0,1,0 p=384)	13940.93	93.38	123.23	29.85	Model based on acf & pacf estimations
(1,0,2) (0,1,0 p = 384)	14148.26	103.29	130.51	27.22	auto.arima
(1,1,1)	18044.66	90.24	183.61	93.37	Baseline

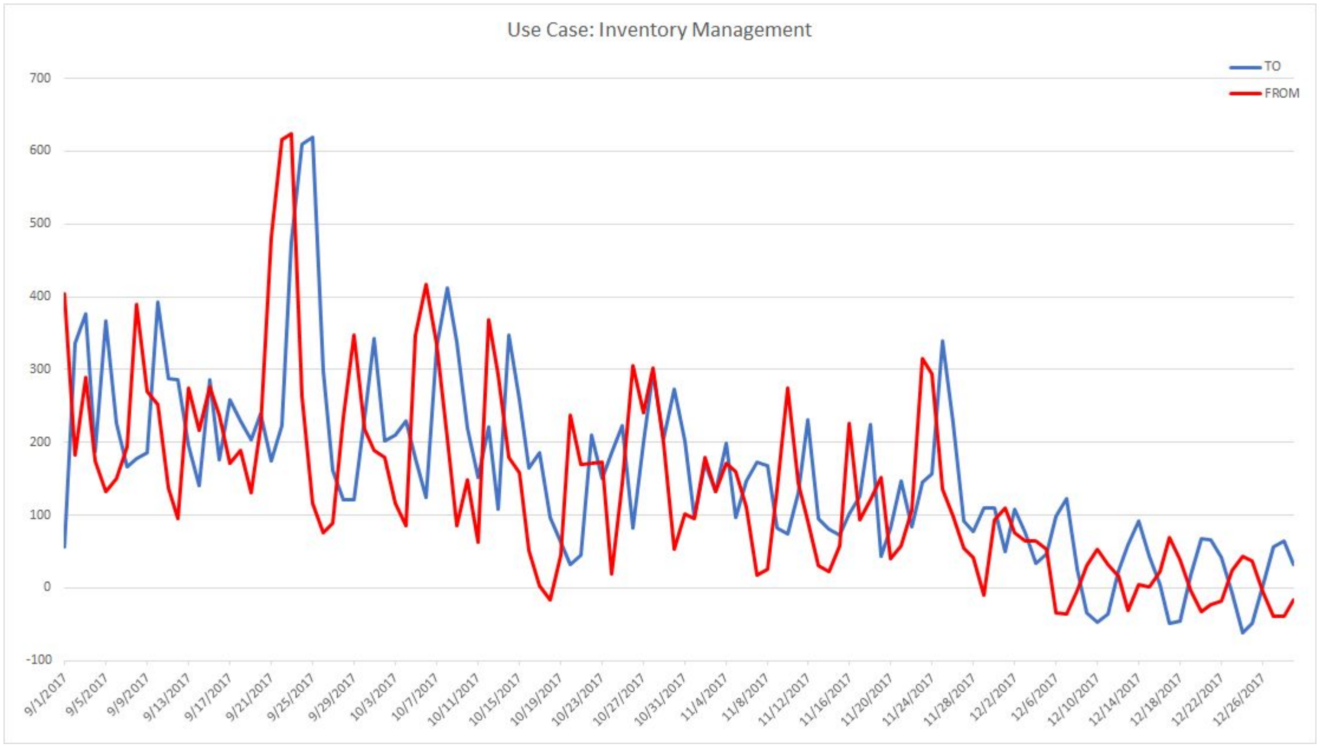
Interestingly enough although the auto.arima model recommended a lower p value compared to the FROM data, the best fit model for the TO data ended up being the same as the FROM data best fit model of **(4,1,4) (0,1,0 p=384).** The table listed above only includes top 3 performing "random" models compared to the models which were mentioned in the previous sections, for a full list of the models which were used please see the appendix.



Cross validation "leave one out"

Although out of scope for the purposes of this capstone the next step would be to use leave one out cross validation (LOOCV) where one point is repeatedly left out of the data set and tested in order to cross-validate in both the TO and FROM datasets.

Conclusion,Use Case & Model Improvements

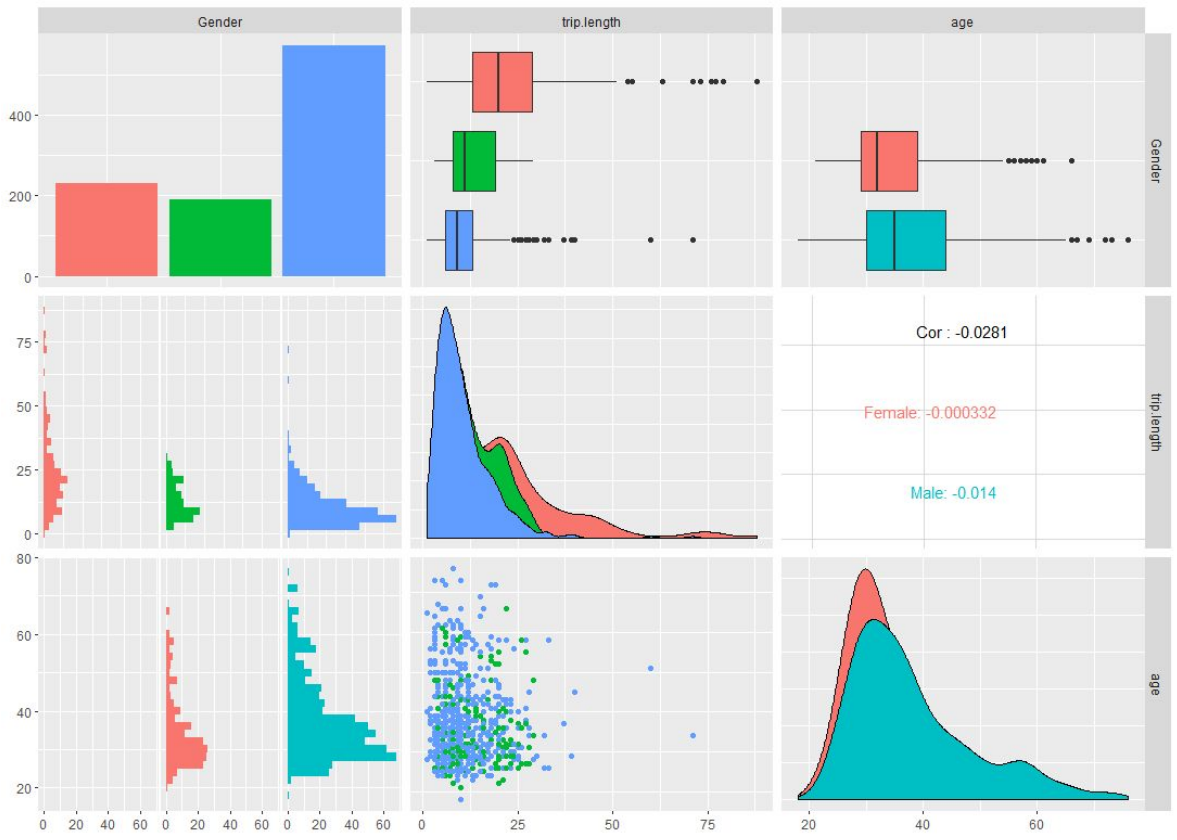


The graph above depicts the forecasts for TO and FROM. For a business use-case this visualization could help inventory management identify when the demand and supply gap is the highest. For example October 8th & 9th, where the number of bikes taken to the station and from the station vary significantly as depicted by the distance between the TO and FROM lines above, would be 2 days where the demand supply gap varies significantly.

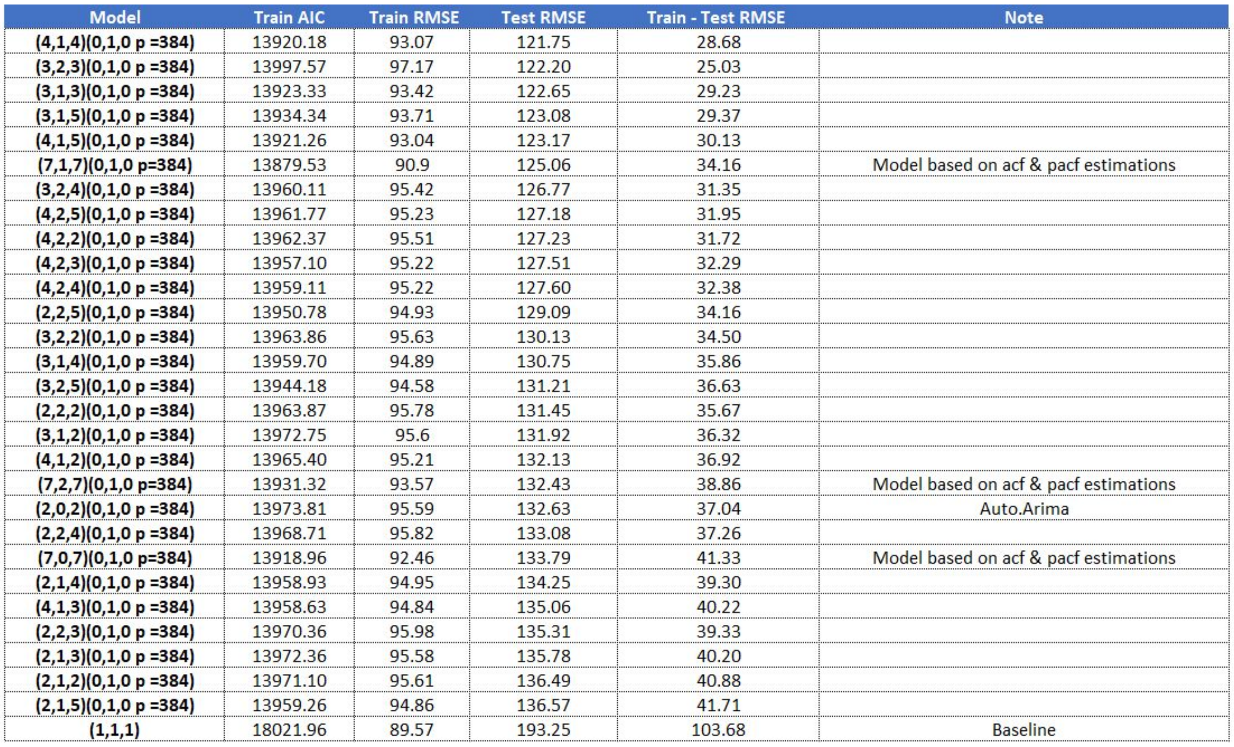
Continuous improvements to these models are key in order to improve the accuracy of forecasts so that all Divvy Bike users always have a bike when they need it. It would be of interest to collect information on why this data set had missing days. One scenario for these missing days is that the sensors recording trips were malfunctioning. A second scenario would be that no trips were taken and therefore there was no start and stop time in the data set. For the purposes of this project I took the mean from the previous and following day for the days which did not have any recorded observations (there were about 99 days added to the TO and FROM data sets each) but receiving this background information on the missing days could possibly change the way missing days were handled. Another improvement to these models would be transforming a forecast so that TO and FROM do not go into the negative since in reality, there can never be negative bikes taken to or from the station. One way to rectify this issue would be possibly using a different model other than arima, which is known for this issue. Forecasting on an hourly or even minute by minute basis would also improve forecasting accuracy even further.

Appendix

GGPAIRS

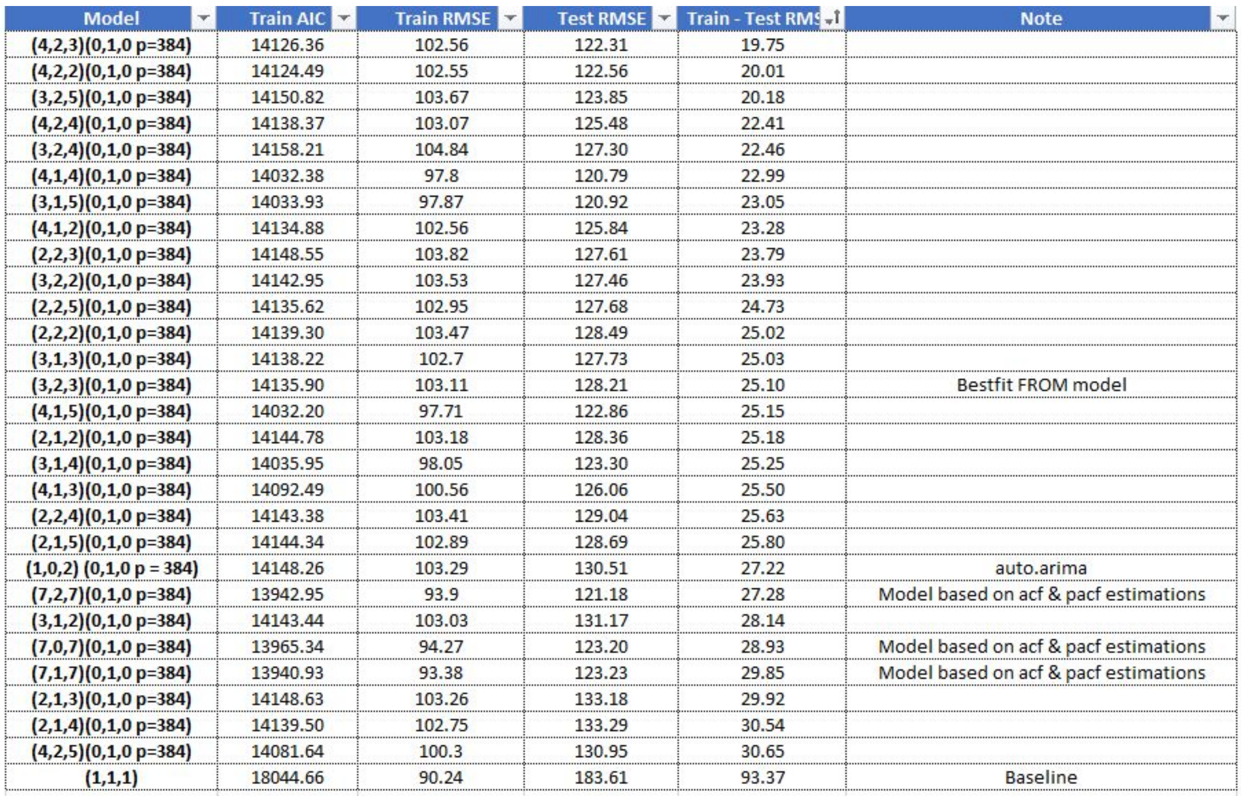


FROM Comprehensive Model Chart



Model	Train AIC	Train RMSE	Test RMSE	Train - Test RMSE	Note
(4,1,4)(0,1,0 p =384)	13920.18	93.07	121.75	28.68	
(3,2,3)(0,1,0 p =384)	13997.57	97.17	122.20	25.03	
(3,1,3)(0,1,0 p =384)	13923.33	93.42	122.65	29.23	
(3,1,5)(0,1,0 p =384)	13934.34	93.71	123.08	29.37	
(4,1,5)(0,1,0 p =384)	13921.26	93.04	123.17	30.13	
(7,1,7)(0,1,0 p=384)	13879.53	90.9	125.06	34.16	Model based on acf & pacf estimations
(3,2,4)(0,1,0 p =384)	13960.11	95.42	126.77	31.35	
(4,2,5)(0,1,0 p =384)	13961.77	95.23	127.18	31.95	
(4,2,2)(0,1,0 p =384)	13962.37	95.51	127.23	31.72	
(4,2,3)(0,1,0 p =384)	13957.10	95.22	127.51	32.29	
(4,2,4)(0,1,0 p =384)	13959.11	95.22	127.60	32.38	
(2,2,5)(0,1,0 p =384)	13950.78	94.93	129.09	34.16	
(3,2,2)(0,1,0 p =384)	13963.86	95.63	130.13	34.50	
(3,1,4)(0,1,0 p =384)	13959.70	94.89	130.75	35.86	
(3,2,5)(0,1,0 p =384)	13944.18	94.58	131.21	36.63	
(2,2,2)(0,1,0 p =384)	13963.87	95.78	131.45	35.67	
(3,1,2)(0,1,0 p =384)	13972.75	95.6	131.92	36.32	
(4,1,2)(0,1,0 p =384)	13965.40	95.21	132.13	36.92	
(7,2,7)(0,1,0 p=384)	13931.32	93.57	132.43	38.86	Model based on acf & pacf estimations
(2,0,2)(0,1,0 p =384)	13973.81	95.59	132.63	37.04	Auto.Arima
(2,2,4)(0,1,0 p =384)	13968.71	95.82	133.08	37.26	
(7,0,7)(0,1,0 p=384)	13918.96	92.46	133.79	41.33	Model based on acf & pacf estimations
(2,1,4)(0,1,0 p =384)	13958.93	94.95	134.25	39.30	
(4,1,3)(0,1,0 p =384)	13958.63	94.84	135.06	40.22	
(2,2,3)(0,1,0 p =384)	13970.36	95.98	135.31	39.33	
(2,1,3)(0,1,0 p =384)	13972.36	95.58	135.78	40.20	
(2,1,2)(0,1,0 p =384)	13971.10	95.61	136.49	40.88	
(2,1,5)(0,1,0 p =384)	13959.26	94.86	136.57	41.71	
(1,1,1)	18021.96	89.57	193.25	103.68	Baseline

TO Comprehensive Model Chart



Model	Train AIC	Train RMSE	Test RMSE	Train - Test RMS	Note
(4,2,3)(0,1,0 p=384)	14126.36	102.56	122.31	19.75	
(4,2,2)(0,1,0 p=384)	14124.49	102.55	122.56	20.01	
(3,2,5)(0,1,0 p=384)	14150.82	103.67	123.85	20.18	
(4,2,4)(0,1,0 p=384)	14138.37	103.07	125.48	22.41	
(3,2,4)(0,1,0 p=384)	14158.21	104.84	127.30	22.46	
(4,1,4)(0,1,0 p=384)	14032.38	97.8	120.79	22.99	
(3,1,5)(0,1,0 p=384)	14033.93	97.87	120.92	23.05	
(4,1,2)(0,1,0 p=384)	14134.88	102.56	125.84	23.28	
(2,2,3)(0,1,0 p=384)	14148.55	103.82	127.61	23.79	
(3,2,2)(0,1,0 p=384)	14142.95	103.53	127.46	23.93	
(2,2,5)(0,1,0 p=384)	14135.62	102.95	127.68	24.73	
(2,2,2)(0,1,0 p=384)	14139.30	103.47	128.49	25.02	
(3,1,3)(0,1,0 p=384)	14138.22	102.7	127.73	25.03	
(3,2,3)(0,1,0 p=384)	14135.90	103.11	128.21	25.10	Bestfit FROM model
(4,1,5)(0,1,0 p=384)	14032.20	97.71	122.86	25.15	
(2,1,2)(0,1,0 p=384)	14144.78	103.18	128.36	25.18	
(3,1,4)(0,1,0 p=384)	14035.95	98.05	123.30	25.25	
(4,1,3)(0,1,0 p=384)	14092.49	100.56	126.06	25.50	
(2,2,4)(0,1,0 p=384)	14143.38	103.41	129.04	25.63	
(2,1,5)(0,1,0 p=384)	14144.34	102.89	128.69	25.80	
(1,0,2) (0,1,0 p = 384)	14148.26	103.29	130.51	27.22	auto.arima
(7,2,7)(0,1,0 p=384)	13942.95	93.9	121.18	27.28	Model based on acf & pacf estimations
(3,1,2)(0,1,0 p=384)	14143.44	103.03	131.17	28.14	
(7,0,7)(0,1,0 p=384)	13965.34	94.27	123.20	28.93	Model based on acf & pacf estimations
(7,1,7)(0,1,0 p=384)	13940.93	93.38	123.23	29.85	Model based on acf & pacf estimations
(2,1,3)(0,1,0 p=384)	14148.63	103.26	133.18	29.92	
(2,1,4)(0,1,0 p=384)	14139.50	102.75	133.29	30.54	
(4,2,5)(0,1,0 p=384)	14081.64	100.3	130.95	30.65	
(1,1,1)	18044.66	90.24	183.61	93.37	Baseline