CSP571 - Data Preparation and Analysis

DIVVY BIKE SHARE ANALYSIS

Final Report

Akhil Suryadevara	(A20391322)
Ashwin Kadammaje Giridhar	(A20399458)
Azlagiavanan Senthil	(A20398151)
Poojitha Bangalore Srinivasan	(A20405615)

Table of Contents

ABSTRACT……3

1. PROBLEM DEFINITION…………………………………………………………………………………………..3

1.1 Introduction……………………………………………………………………………………………………….3

1.2 Problem Statement……………………………………………………………………………………………3

1.3 Goal………..3

2. DATA PREPATATION……………………………………………………………………………………………4

3. DATA PREPROCESSING………………………………………………………………………………………5

3.1 Filling in missing values ………………………………………………………………………………………5

3.2 Handling date and time ……………………………………………………………………………………5

3.3 Dropping off unused levels………………………………………………………………………………5

4. EXPLANATORY DATA ANALYSIS OF 2017 DATA………………………………………………………5

4.1 Analysis on user type and gender variables……………………………………………………5

4.2 Analysis on user type and time variables…………………………………………………………6

4.3 Plotting of top 10 best used stations…………………………………………………………………7

4.4 Plotting of top 10 least used stations………………………………………………………………8

4.5 Visualization of the top 10 best and least used stations in a map…………………………8

4.6 Finding and plotting the frequency of rides for all the days in a week in the year 2017….9

5. TIME SERIES ANALYSIS……………………………………………………………………………………10

5.1 Generating a time series using ts function and plotting it…………………………………10

5.2 Decomposing time series………………………………………………………………………………10

5.3 Plotting of the decomposed time series……………………………………………………………11

5.4 Computing acf for the time series……………………………………………………………………11

5.5 Computing pacf for the time series…………………………………………………………………12

5.6 Generating ARIMA model for the obtained time series………………………………………13

5.7 PLOTTING ARIMA(1,2,7) MODEL…………………………………………………………………………13

5.8 Forecasting and plotting the obtained model……………………………………………………13

6. CONCLUSION……………………………………………………………………………………………………14

7.FUTUREWORK……………………………………………………………………………………………………14

8. REFERENCES……………………………………………………………………………………………………15

9. APPENDIX……15

9.1 Source code…………………………………………………………………………………………………….15

ABSTRACT

Bicycle-sharing systems, which can provide shared bike usage services for the public, have been launched in many big cities. In bicycle-sharing systems, people can borrow and return bikes at any stations in the service region very conveniently. Therefore, bicycle-sharing systems are normally used as a short- distance trip supplement for private vehicles as well as regular public transportation. Meanwhile, for stations located at different places in the service region, the bike usages can be quite skewed and imbalanced. Some stations have too many incoming bikes and get jammed without enough docks for upcoming bikes, while some other stations get empty quickly and lack enough bikes for people to check out. Therefore, inferring the potential destinations and arriving time of each individual trip beforehand can effectively help the service providers schedule manual bike re-dispatch in advance. In this project, we will study the prediction of bide rides for the future through some analysis of the previous data and times series modelling. To address the problem, we study a real-world bicycle-sharing system - DIVVY.

1. PROBLEM DEFINITION

1.1 Introduction

Biking is beneficial to both of our health and environment. Many cities have launched public bike-sharing system to promote bicycle usage. In bike-sharing systems, people can borrow bikes from one station and return the bike back to any stations in the city, which can be used as a short-distance trip supplement for private vehicles as well as regular public transportation. Divvy program in Chicago is one of the mature systems that we would like to explore. Divvy is a bicycle sharing system in the City of Chicago operated by 'Motivate' for the Chicago Department of Transportation. It operates 5800 bicycles at 580 stations. Divvy is a fun and affordable way to get around Chicago with a big customer base. In 2015 approximately 3.2 million trips were made.

In bike-share system, the travel behaviors of people in different age and gender groups can be quite different. Meanwhile, for stations located at different places in the city, the bike usage can be quite skewed and unbalanced. Some stations that individuals like to borrow bikes from will lack enough bikes for people to check out, while some other stations that people normally return the bikes to will get jammed easily without enough docks for upcoming bikes. Therefore, we want to discover different usage patterns among various user groups and time periods, as well as flow patterns of stations. The goal of our project is to identify patterns in Divvy bicycle usage to benefit Divvy's business operation.

1.2 Problem Statement

To identify patterns in Divvy bicycle usage to benefit Divvy's business operation. The main objective of the project is to determine the number of bike rides for the future.

1.3 Goal

To build a time series regression model using Auto Regressive Integrated Moving Average method(ARIMA)

Forecasting and plotting of the obtained model for the upcoming year(2018)

2. **DATA PREPATATION**

Divvy was launched in late June 2013. The system had 474 stations and 8274 bikes in 2015, and increased to 535 stations and 9214 bikes in 2016, which greatly improved its coverage and maximum working load. All bikes are available 24 hours, each station has a touch screen kiosk and docking system which support bikes check-in and check-out using a member key or ride code. Bike-sharing system allows people to borrow bikes with either "24-hour pass" or "annual subscribed membership". "24-hour pass" is usually preferred by people for temporary usage, such as tourists and occasional riders, but charges per day are slightly higher. Meanwhile, "annual subscribed membership" is a great deal for people with frequent travel needs, such as local office workers and students.

The data we used in this project was obtained from Divvy website (https://www.divvybikes.com) The datasets include trip, station and customer type information from January 2015 to December 2017. Trip dataset has observations described by trip ID, start time and station, stop time and station and trip duration. Station dataset consists of station name, dock capacity, coordinates and created time.

Each row in the dataset is recorded from an individual's usage with the below columns

Variable Name	**Description**
trip_id	ID attached to each trip taken
starttime	day and time trip started
stoptime	day and time trip ended
bikeid	Id attached to each bike
tripduration	time of trip in seconds
from_station_id	ID of station where trip originated
from_station_name	name of station where trip originated
to_station_id	ID of station where trip terminated
to_station_name	name of station where trip terminated
usertype	"Customer" is a rider who purchased a 24-hour pass, "Subscriber" is a rider who purchased an annual membership
gender	gender of rider
birthyear	birth year of rider.

Table 1: Variable names and description for Divvy_trips data

Variable Name	**Description**
id	ID attached to each trip as in trips data
name	Name of the station
latitude	Gives the latitude of the station
longitude	Gives the longitude of the station
dpcapacity	Dock capacity in each station
dateCreated	Date when the station was created

Table 2: Variable names and description for Divvy_stations data

3. DATA PREPROCESSING

The dataset we use is incomplete and not clean. Hence some of the techniques followed in the project are:

- Filling in missing values
- Handling date and time
- Dropping off unused levels

3.1 Filling in missing values

Variables such as gender and birth year have the missing values.
Fill in with 'Unknown' for the missing values in variable gender.
For variable birth year, fill in with the mean value for it.

3.2 Handling date and time

In order to achieve standard data formats and make further statistical analysis easier, we unified the time format. It was efficiently handled using POSIXt function with the following format ('%m%d%y %H%M')

3.3 Dropping off unused levels

Among the subscriber, customer and dependent we are dropping levels of dependent as it has very less instance values from the variable user type.

4. EXPLANATORY DATA ANALYSIS OF 2017 DATA

There are several interesting takeaways we can draw from our experimental results.
Following are some of the analysis done on the 2017 data before performing time series analysis:

- Analysis on user type and gender variables
- Plotting of top 10 best used stations.
- Plotting of top 10 least used stations.
- Visualization of the top 10 best and least used stations in a map.
- Finding and plotting the frequency of rides for all the days in a week in the year 2017.
- Finding the frequency of rides throughout the year.

4.1 Analysis on user type and gender variables

To predict the trips taken by users in bicycle-sharing systems, we need to understand the composition of people using the bikes at first. By studying the historical trip record data, we count up the numbers of trips taken by "customers" and "subscribers" respectively among the Divvy users and the statistical results.

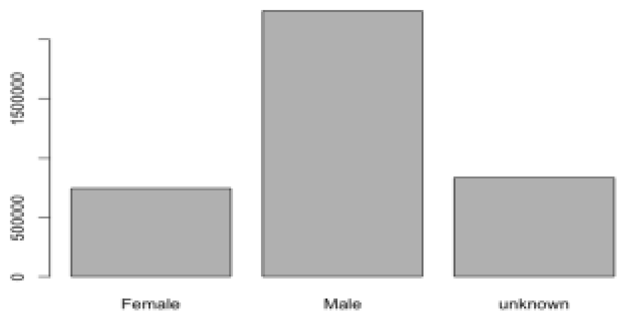

Figure 1

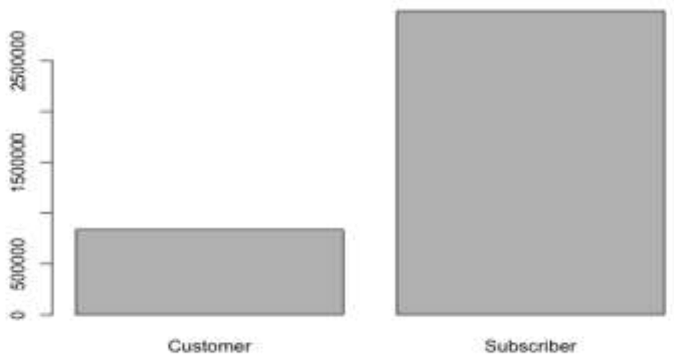

Figure 2

- Figure 1 plot shows the total number of user types based on the gender.
- Figure 2 plot shows the number of customers and subscribers after dropping level ‘dependent’.

4.2 Analysis on user type and time variables

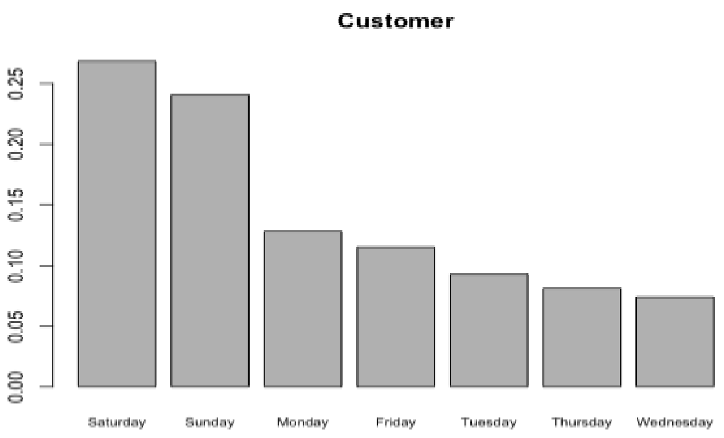

Figure 3 showing the frequency of rides for customers.

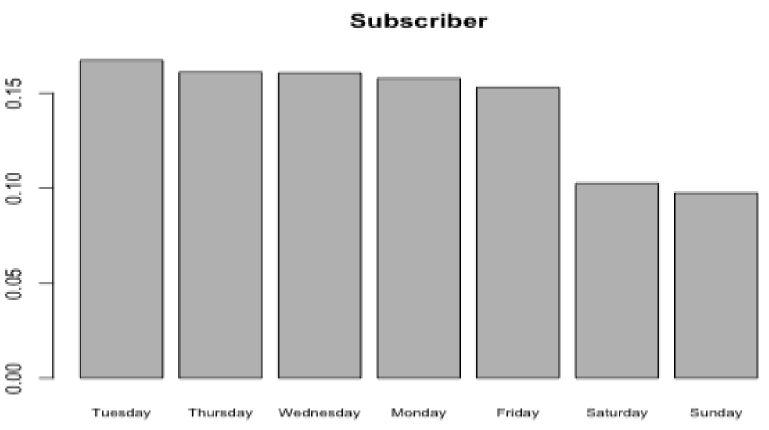

Figure 4 showing the frequency of rides for subscribers.

- The graphs show the number of rides done by customers and subscribers on a weekly basis in the increasing order.
- Based on the whole dataset, we show the number of trips finished by customers and subscribers respectively on each weekday from Sunday to Saturday. Based on the results, we observe that "customers" bike usage pattern is totally different from that of "subscribers". "Subscribers" mainly use the Divvy bike during the weekdays from Monday to Friday, and their usages on weekends (i.e., Sunday and Saturday) drop a lot whereas "customers" use them a lot on weekends.

4.3 Plotting of top 10 best used stations.

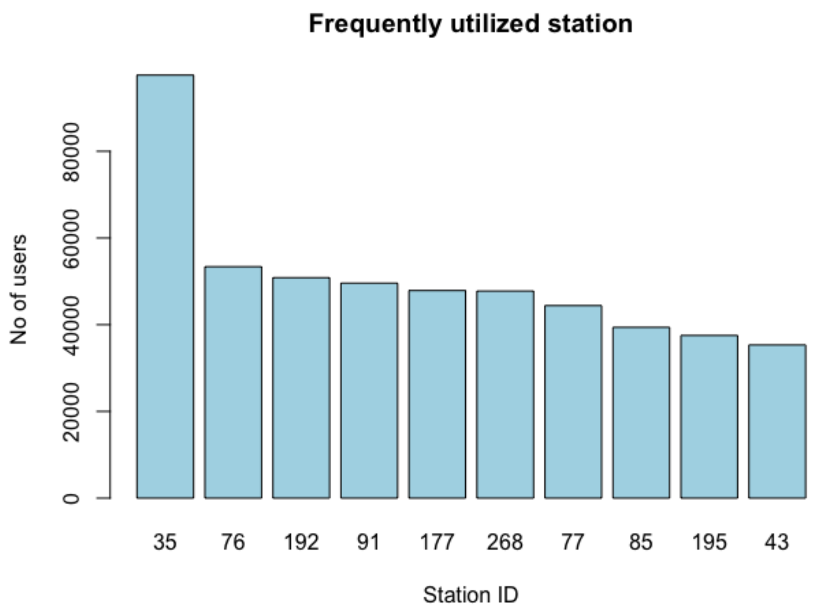

Figure 5

- The bar plot shows the frequently used station with No. of users vs Station ID.
- Among the station IDs, station 35 i.e. 'Streeter and Dr Ave' is the best used station.

4.4 Plotting of top 10 least used stations.

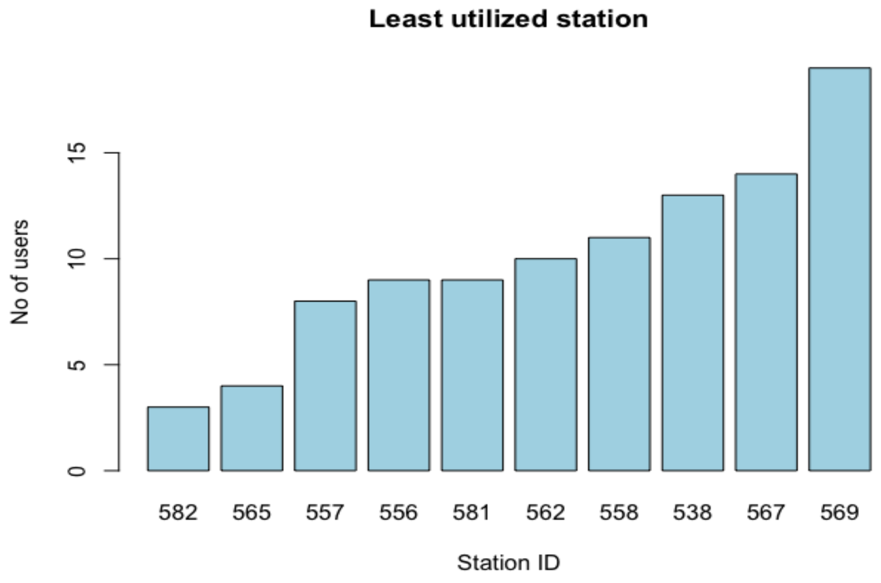

Figure 6

- The bar plot shows the least used station with No. of users vs Station ID.
- Among the station IDs, station 582 i.e. 'Philip Ave and 82nd Street' is the least used station.

4.5 Visualization of the top 10 best and least used stations in a map.

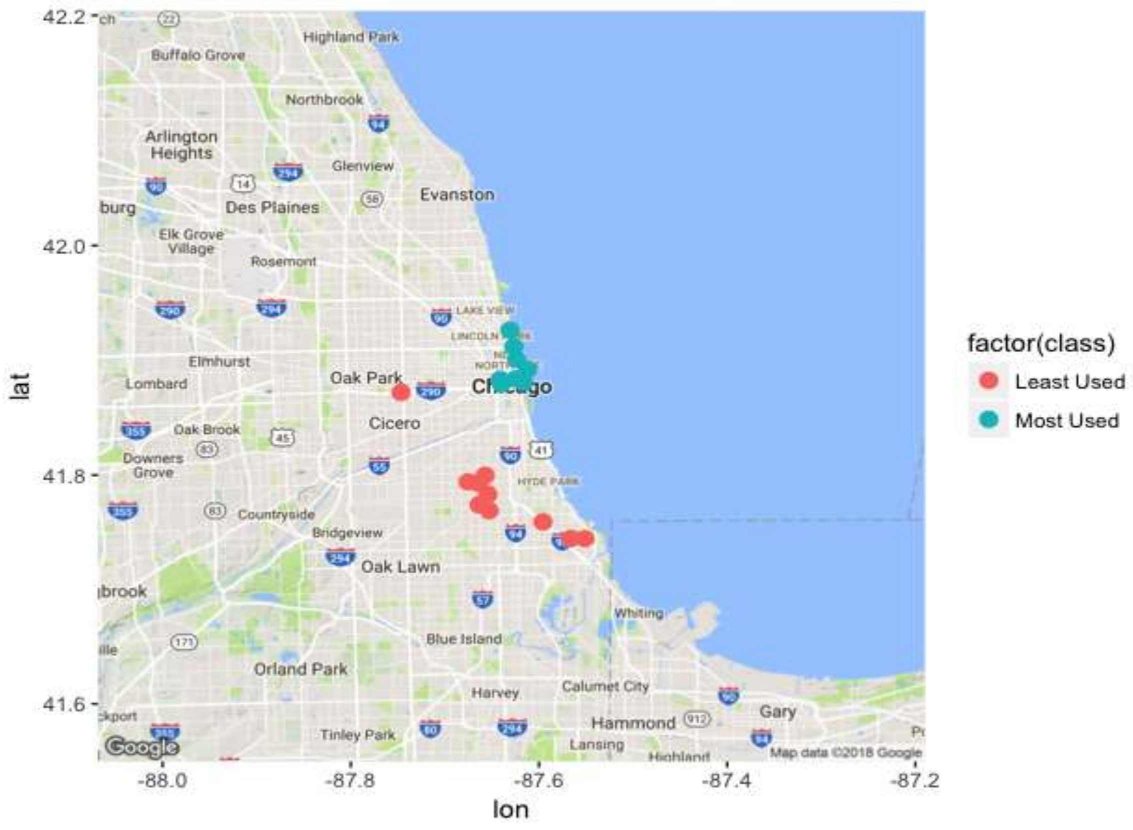

Figure 7

- Bind Divvy trip's data to the Divvy station's data and generate 2 points which matches the top 10 best and the least frequently used stations.

- Bind these 2 points using rbind function and using ggmap function plot them in a map using getmap.

4.6 Finding and plotting the frequency of rides for all the days in a week in the year 2017

To predict the number of bide rides, the frequency of bike rides for each day of the week is also important .Hence studying them using the Divvy_stations data we obtain the following results:

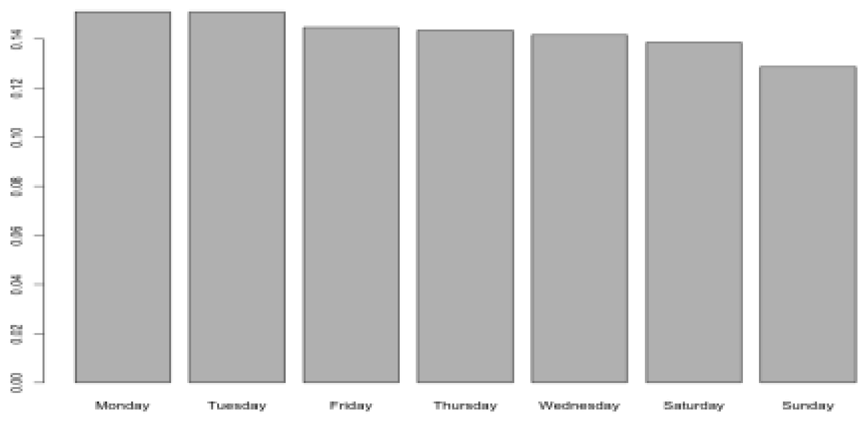

Figure 8

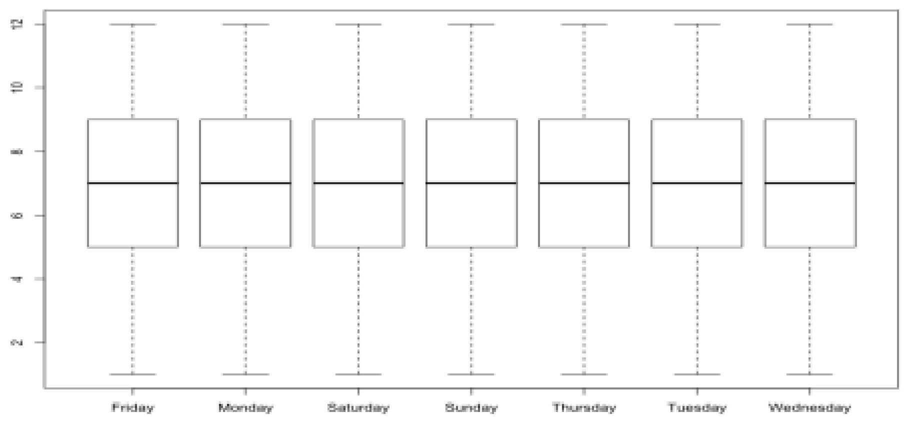

Figure 9

- The bar plot in Figure 8, shows the average no. of rides for each day in a week in the increasing order.
- The plot on the Figure 9, is obtained by comparing the previous output with the frequency of the starting day of the month in that year in an increasing order. Therefore, bikes used on Fridays are comparatively higher.

5. TIME SERIES ANALYSIS

Following are the steps done for the time series analysis:

- Generating a time series using ts function and plotting the same.
- Decomposing time series.

- Plotting of the decomposed time series.
- Computing acf and pacf for the time series.
- Generating ARIMA model for the obtained time series.
- Forecasting and plotting the obtained model.

5.1 Generating a time series using ts function and plotting it

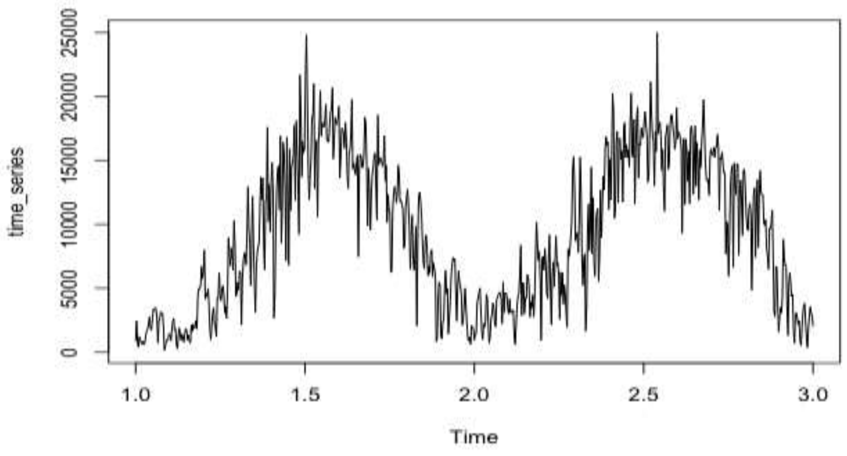

Figure 10

- The function ts is used to create time series object. On applying this function, we generate time series and plot it.
- ts function just predicts the time series for the objects present and does not forecast time series object.

5.2 Decomposing time series.

The decomposition of time series is a statistical task that deconstructs a time series into several components, each representing one of the underlying categories of patterns.

- Decomposition based on rates of change:

  This is the technique which we have used because it does seasonal adjustments. In this method the time series are decomposed into:

- T_{t}, the trend component at time t, which reflects the long-term progression of the series . A trend exists when there is a persistent increasing or decreasing direction in the data.
- S_{t}, the seasonal component at time t, reflecting seasonality. A seasonal pattern exists when a time series is influenced by seasonal factors. Seasonality occurs over a fixed and known period.

5.3 Plotting of the decomposed time series.

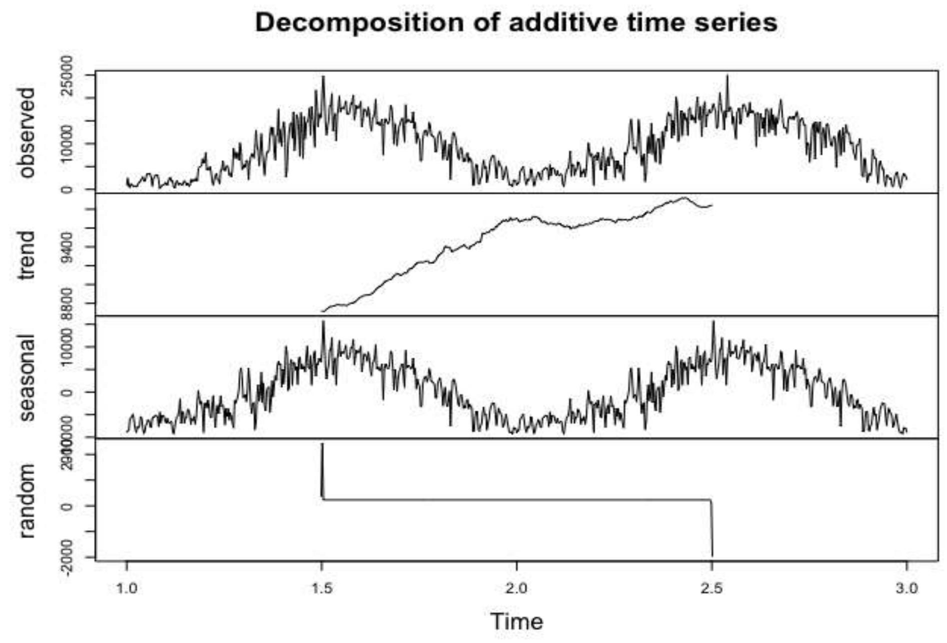

Figure 11

The components in the graph in Figure 11, are defined as follows:

- Trend – An increasing or decreasing value in the series.
- Seasonal – The repeating short term cycle in the series.
- Noise – The random variation in the series.

It can be seen from the graph that observed component and seasonal component are pretty much the same with the trend component increasing which makes obvious that the time series prediction depends on the seasonal component.

5.4 Computing acf for the time series.

The ACF property defines a distinct pattern for the autocorrelations. For a positive value of ϕ1, the ACF exponentially decreases to 0 as the lag h increases. For negative ϕ1, the ACF also exponentially decays to 0 as the lag increases

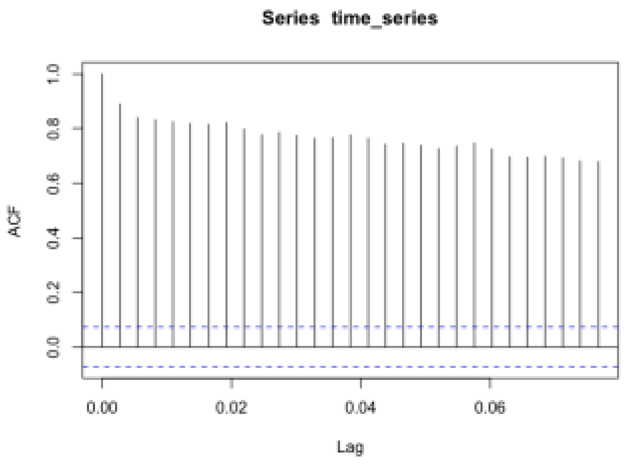

Figure 12

- From the obtained acf graph in Figure 12, we can see that the time series is non stationary.
- Hence acf is decaying or decreasing very slowly well above the significance range(dotted blue lines).
- This proves that the series is non stationary.

5.5 Computing pacf for the time series.

In time series analysis, the partial autocorrelation function (PACF) gives the partial correlation of a time series with its own lagged values, controlling for the values of the time series at all shorter lags. It contrasts with the autocorrelation function, which does not control for other lags.

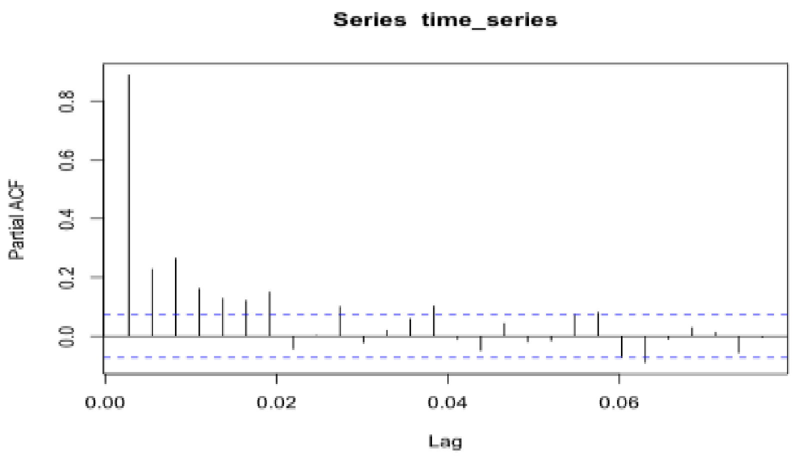

Figure 13

- From the obtained pacf graph in Figure 13, we can see that there are some lags contracting the acf graph.
- Hence counting the lags in pacf to determine the order of the moving time series model.

5.6 Generating ARIMA model for the obtained time series.

- What is ARIMA ?
  - ARIMA stands for Autoregressive Integrated Moving Average models.
  - Univariate (single vector) ARIMA is a forecasting technique that projects the future values of a series based entirely on its own inertia.
  - Its main application is in the area of short term forecasting requiring at least 40 historical data points.
  - It works best when your data exhibits a stable or consistent pattern over time with a minimum amount of outliers.
- Determining the parameters for the model(p,d,q)
  - p refers to the number of autoregressive terms.
  - d refers to the degree of differencing.
  - q refers to the number of lagged forecast errors.
  - The model is valuated on the obtained aic or bic value.

5.7 PLOTTING ARIMA(1,2,7) MODEL

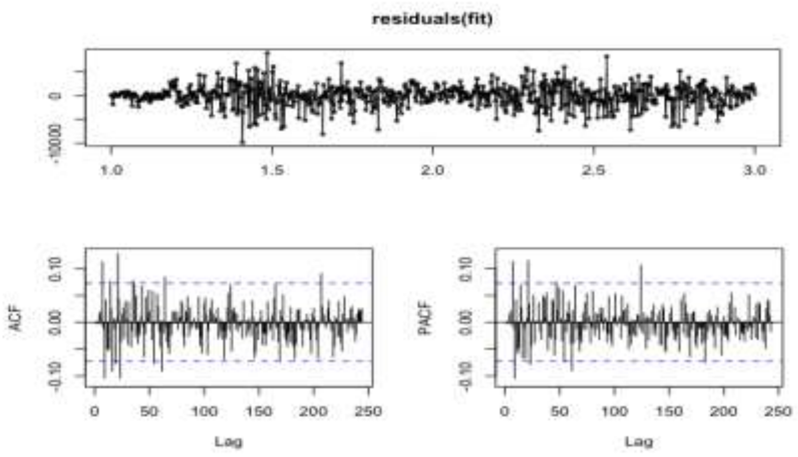

Figure 14

- From the lags in acf and pacf obtained in the Figure 14, we can see that there is a lag at 7.
- Hence we can see that the ARIMA model fits better when q=7 and there are no much significant lags after q=7.
- The residuals obtained also shows small error range when compared to other parameter values.

5.8 Forecasting and plotting the obtained model

Using forecast function and the obtained ARIMA (1,2,7) model we can predict how the model performs in the future

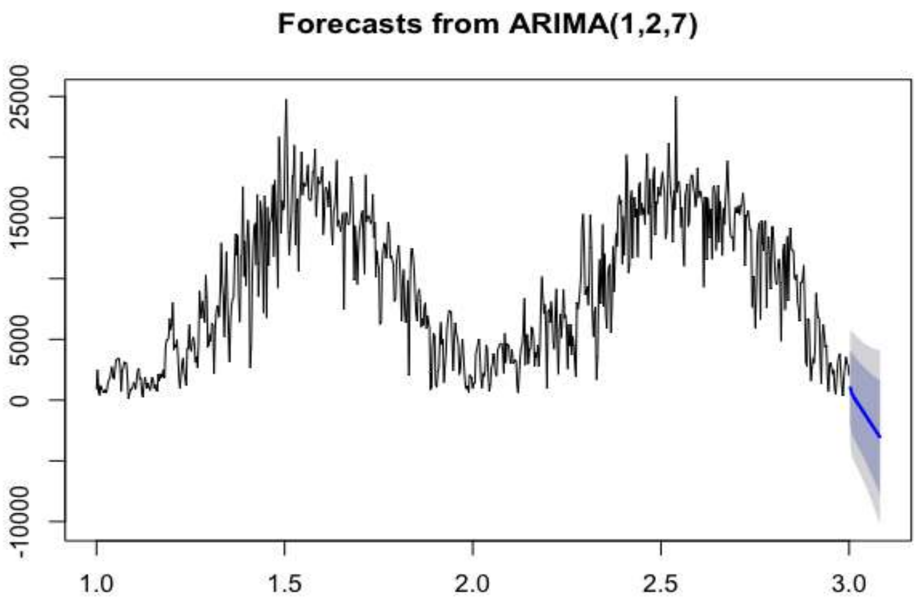

Figure 15

- From the obtained graph in Figure 15 we can see that the forecast is decreasing.
- Hence from the above graph we can forecast that the number of bike rides for the upcoming year is initially decreasing.

6. CONCLUSION

We conclude that the plotted predictions are based on the assumptions that there will be no other seasonal fluctuations in the data and the change in number of bicycles from one day to another is more or less constant in mean and variance. Here in our forecasted model, the values may be overfitted because the q parameter (number of lagged forecast errors) taken for our arima model is pretty high. But according to performance metric AIC or BIC for the ARIMA, this value of q i.e. q=7 performs better.

7. FUTURE WORK:

We plan to continue to develop on the ideas presented in this project in a few ways. Firstly, we would like to implement the time series using a part of the data , say the first six months of the 2015 data and implement time series on that data and slide the window for the rest and obtain the forecast for the future so that overfitting of data can be avoided. Secondly, we would like to do some more data analysis like finding the distance travelled for a bike ride using the from and to station id on the dataset and also perform clustering on the data using some clustering techniques like K-means and Hierachical Clustering, so that if there is a shortage of bikes in a particular station, using the nearest distance it would be easy to bring in bikes to that station.

8. REFERENCES:

[1] Bicycle-Sharing System Analysis and Trip Prediction-https://arxiv.org/pdf/1604.00664.pdf.
[2] Redistribution of the Chicago's Divvy Bike-Share Stations-
https://scholarworks.wmich.edu/cgi/viewcontent.cgi?referer=https://www.google.com/&httpsredir=1&article=1724&context=masters_theses.
[3] Introduction to Forecasting with ARIMA in R -
https://www.datascience.com/blog/introduction-to-forecasting-with-arima-in-r-learn-data-science-tutorials.
[4] An Introductory study on Time Series Modelling and Forecasting -
https://arxiv.org/pdf/1302.6613.pdf.
[5] The use of ARIMA for prediction of Incidence of Dysentry -
http://journals.sagepub.com/doi/abs/10.1177/1010539516645153.
[6] ARIMA - https://en.wikipedia.org/wiki/Autoregressive_integrated_moving_average.
[7] *Asteriou, Dimitros; Hall, Stephen G. (2011). "ARIMA Models and the Box–Jenkins Methodology". Applied Econometrics (Second ed.). Palgrave MacMillan. pp. 265–286. ISBN 978-0-230-27182-1*

9. APPENDIX:

8.1Source code

```
rm(list=ls())

#list of libraries:

library(data.table)
library(plyr)

library(lubridate)

library(ggplot2)

library(graphics)

library(ggmap)

library(devtools)

library(forecast)
```

#Reading the dataset to a data table

```
F1 <- fread(file = "/Users/alagu/Desktop/Divvy dataset/Divvy_Trips_2017_Q1Q2/Divvy_Trips_2017_Q1.csv",
header = TRUE)

View(F1)

F2 <- fread(file = "/Users/alagu/Desktop/Divvy dataset/Divvy_Trips_2017_Q1Q2/Divvy_Trips_2017_Q2.csv",
header = TRUE)

View(F2)

F3 <- fread(file = "/Users/alagu/Desktop/Divvy dataset/Divvy_Trips_2017_Q3Q4/Divvy_Trips_2017_Q3.csv",
header = TRUE)

View(F3)

F4 <- fread(file = "/Users/alagu/Desktop/Divvy dataset/Divvy_Trips_2017_Q3Q4/Divvy_Trips_2017_Q4.csv",
header = TRUE)

View(F4)

divvy_list <- list(F1,F2,F3,F4)

divvy <- ldply(divvy_list, data.table)

class(divvy)

View(divvy)
```

#Finding the structure of the object "divvy" using str and generating the summary for divvy.

```
str(divvy)

summary(divvy)
```

# creating a vector for numerical values from the dataset for future usage in the code.

```
numeric_list <- c('trip_id', 'bikeid', 'tripduration', 'from_station_id', 'to_station_id', 'birthyear')

for (i in numeric_list) {

  divvy[,i] = as.integer(divvy[,i])

}
```

#Pre-processing.Handling missing values.

#Variables gender and birthyear have missing values.

```
divvy$gender[divvy$gender==""] <- 'unknown'

boxplot(divvy$birthyear)

divvy$birthyear[is.na(divvy$birthyear)] <- 1985
```

#handling date-time using POSIXt Date-time Class.

```
divvy$start_time = as.POSIXlt(divvy$start_time,format="%m/%d/%Y %H:%M")

divvy$end_time = strptime(divvy$end_time, format = "%m/%d/%Y %H:%M")


divvy$usertype = as.factor(divvy$usertype)

divvy$gender = as.factor(divvy$gender)


divvy <- divvy[!(divvy$usertype=='Dependent'),]

divvy$usertype <- droplevels(divvy$usertype)


str(divvy)

summary(divvy)

divvy <- divvy[!(divvy$birthyear < 1940),]


plot(divvy$usertype)

plot(divvy$gender)

hist(divvy$birthyear)


boxplot(divvy$birthyear)


divvy_final <- divvy

View(divvy_final)


divvy_final$start_hour <- hour(divvy_final$start_time)

divvy_final$start_month <- month(divvy_final$start_time)

str(divvy_final)


hist(divvy_final$start_hour)

hist(divvy_final$start_month)


divvy_final$week <- weekdays(divvy_final$start_time)
```

```
divvy_final$week <- as.factor(divvy_final$week)


barplot(sort(prop.table(table(divvy_final$week)), decreasing = TRUE))


plot(divvy_final$week, divvy_final$start_month)


divvy_subscriber <- divvy_final[divvy_final$usertype=='Subscriber',]
class(divvy_subscriber)


divvy_customer <- divvy_final[divvy_final$usertype=='Customer',]
class(divvy_customer)


str(divvy_subscriber)
summary(divvy_subscriber)
View(divvy_subscriber)


str(divvy_customer)
summary(divvy_customer)
View(divvy_customer)


par(mfrow= c(1,2))
barplot(sort(prop.table(table(divvy_subscriber$week)), decreasing = TRUE), main = 'Subscriber', cex.names = 0.7)
barplot(sort(prop.table(table(divvy_customer$week)), decreasing = TRUE), main = 'Customer', cex.names = 0.7)
par(mfrow = c(1,1))


str(as.factor(divvy_final$from_station_id))


barplot(sort(table(as.factor(divvy_final$from_station_id)), decreasing = TRUE)[1:10], main = 'Frequently utilized
station', xlab = "Station ID", ylab = 'No of users', col = 'lightblue')
best_station_id <- sort(table(as.factor(divvy_final$from_station_id)), decreasing = TRUE)[1:10]
names(best_station_id)


best_station <- c()
for(i in names(best_station_id))
{
  best_station[i] = divvy_final$from_station_name[divvy_final$from_station_id==i][1]
}

print(best_station)


barplot(sort(table(as.factor(divvy_final$from_station_id)))[1:10], main = 'Least utilized station', xlab = "Station ID",
ylab = 'No of users', col = 'lightblue')

least_station_id <- sort(table(as.factor(divvy_final$from_station_id)))[1:10]

names(least_station_id)


least_station <- c()

for(i in names(least_station_id))

{

  least_station[i] = divvy_final$from_station_name[divvy_final$from_station_id==i][1]

}

print(least_station)


location <- fread("/Users/alagu/Desktop/Divvy dataset/Divvy_Trips_2017_Q3Q4/Divvy_Stations_2017_Q3Q4.csv",
header = TRUE)

View(location)

location$name <- NULL

location$online_date = NULL

divvy_final <- cbind(divvy_final, location[match(divvy_final$from_station_id, location$id),][,2:4])



point1 <- divvy_final[match(names(best_station_id), divvy_final$from_station_id),][,16:18]

point2 <- divvy_final[match(names(least_station_id), divvy_final$from_station_id),][,16:18]

points <- rbind(point1, point2)

points$class <- c(rep('Most Used',10), rep('Least Used',10))

View(points)

map <- get_map(location = 'chicago', zoom = 10)

ggmap(map) + geom_point(data = points, aes( longitude,latitude, color=factor(class)), size = 3)



table(divvy_final$start_month)


day_wise <- table(as.Date(divvy_final$start_time))

class(day_wise)
```

```
as.data.frame(day_wise)
View(day_wise)
plot(day_wise)

barplot(sort(table(as.factor(divvy_final$from_station_id)), decreasing = TRUE)[1:20], main = 'Frequently utilized station', xlab = "Station ID", ylab = 'No of users', col = 'lightblue')
best_station_id <- sort(table(as.factor(divvy_final$from_station_id)), decreasing = TRUE)[1:20]
names(best_station_id)

best_station <- c()
for(i in names(best_station_id))
{
  best_station[i] = divvy_final$from_station_name[divvy_final$from_station_id==i][1]
}
print(best_station)

barplot(sort(table(as.factor(divvy_final$from_station_id)))[1:20], main = 'Least utilized station', xlab = "Station ID", ylab = 'No of users', col = 'lightblue')
least_station_id <- sort(table(as.factor(divvy_final$from_station_id)))[1:20]
names(least_station_id)

least_station <- c()
for(i in names(least_station_id))
{
  least_station[i] = divvy_final$from_station_name[divvy_final$from_station_id==i][1]
}
print(least_station)
```

#Geographic view

```
points1 <- divvy_final[match(names(best_station_id), divvy_final$from_station_id),][,16:18]
points2 <- divvy_final[match(names(least_station_id), divvy_final$from_station_id),][,16:18]

fullpoints <- rbind(points1, points2)
fullpoints$class <- c(rep('Most Used',20), rep('Least Used',20))
View(fullpoints)
```

```
map <- get_map(location = 'chicago', zoom = 11)

ggmap(map) + geom_point(data = fullpoints, aes(longitude, latitude, color=factor(class)), size = 3)
```

#Time series Analysis:

```
F5 <- fread(file = "/Users/alagu/Desktop/Divvy dataset/Divvy_Trips_2016_Q1Q2/Divvy_Trips_2016_Q1.csv",
header = TRUE)

View(F5)

F6 <- fread(file = "/Users/alagu/Desktop/Divvy dataset/Divvy_Trips_2016_Q1Q2/Divvy_Trips_2016_04.csv",
header = TRUE)

View(F6)

F7 <- fread(file = "/Users/alagu/Desktop/Divvy dataset/Divvy_Trips_2016_Q1Q2/Divvy_Trips_2016_05.csv",
header = TRUE)

View(F7)

F8 <- fread(file = "/Users/alagu/Desktop/Divvy dataset/Divvy_Trips_2016_Q1Q2/Divvy_Trips_2016_06.csv",
header = TRUE)

View(F8)

F9 <- fread(file = "/Users/alagu/Desktop/Divvy dataset/Divvy_Trips_2016_Q3Q4/Divvy_Trips_2016_Q3.csv",
header = TRUE)

View(F9)

F10 <- fread(file = "/Users/alagu/Desktop/Divvy dataset/Divvy_Trips_2016_Q3Q4/Divvy_Trips_2016_Q4.csv",
header = TRUE)

View(F10)

F11 <- fread(file = "/Users/alagu/Desktop/Divvy dataset/Divvy_Trips_2015-Q1Q2/Divvy_Trips_2015_Q1.csv",
header = TRUE)

View(F11)

F12 <- fread(file = "/Users/alagu/Desktop/Divvy dataset/Divvy_Trips_2015-Q1Q2/Divvy_Trips_2015_Q2.csv",
header = TRUE)

View(F12)

F13 <- fread(file = "/Users/alagu/Desktop/Divvy dataset/Divvy_Trips_2015_Q3Q4/Divvy_Trips_2015_Q4.csv",
header = TRUE)

View(F13)

F14 <- fread(file = "/Users/alagu/Desktop/Divvy dataset/Divvy_Trips_2015_Q3Q4/Divvy_Trips_2015_Q7.csv",
header = TRUE)

View(F14)

F15 <- fread(file = "/Users/alagu/Desktop/Divvy dataset/Divvy_Trips_2015_Q3Q4/Divvy_Trips_2015_Q8.csv",
header = TRUE)

View(F15)
```

```
F16 <- fread(file = "/Users/alagu/Desktop/Divvy dataset/Divvy_Trips_2015_Q3Q4/Divvy_Trips_2015_Q9.csv",
header = TRUE)

View(F16)


divvy_list <- list(F1,F2,F3,F4,F5,F6,F7,F8,F9,F10,F11,F12,F13,F14,F15,F16)

divvy <- ldply(divvy_list, data.table)

class(divvy)

View(divvy)


str(divvy)


numeric_list <- c('trip_id', 'bikeid', 'tripduration', 'from_station_id', 'to_station_id', 'birthyear')

for (i in numeric_list) {

  divvy[,i] = as.integer(divvy[,i])

}


divvy$gender[divvy$gender==""] <- 'unknown'

divvy$birthyear[is.na(divvy$birthyear)] <- 1985


divvy$starttime = as.POSIXlt(divvy$starttime,format="%m/%d/%Y %H:%M")

divvy$stoptime = strptime(divvy$stoptime, format = "%m/%d/%Y %H:%M")


divvy$usertype = as.factor(divvy$usertype)

divvy$gender = as.factor(divvy$gender)


summary(divvy)


divvy <- divvy[!(divvy$usertype=='Dependent'),]

divvy$usertype <- droplevels(divvy$usertype)


str(divvy)

summary(divvy)


divvy <- divvy[!(divvy$birthyear < 1940),]


divvy_time <- as.data.frame(table(as.Date(divvy$starttime)))
```

```
View(divvy_time)
plot(divvy_time)
```

#Built in Time series analysis Function(Spectral analysis):

```
time_series <- ts(divvy_time$Freq, frequency = 365)
time_series
plot(time_series)
```

#Decomposing Time series:.

```
time_dec <- decompose(time_series)
plot(time_dec$seasonal)
plot(time_dec$random)
plot(time_dec$trend)
plot(time_dec$figure)
plot(time_dec)
```

#finding autocorrelation function and partial autocorrelation function for the built in model:

```
length(time_series)
par(mfrow=c(2,2))
sapply(1:4, function(x)plot(time_series[-c(731:(731-x+1))],time_series[-c(1:x)]))
par(mfrow=c(2,2))
sapply(7:10, function(x)plot(time_series[-c(731:(731-x+1))],time_series[-c(1:x)]))
acf(time_series)
pacf(time_series)
```

#ARIMA modelling and forecasting:

```
fit <- arima(time_series, order = c(1,2,7))
tsdisplay(residuals(fit))
summary(fit)
AIC(fit)
fcast <- forecast(fit, h=30)
plot(fcast)
summary(fcast)
BIC(fit)
```