Bike Sharing Prediction Problem

By Susiette, Michael, Marco, Ron, Ray

Table of Contents

BUSINESS UNDERSTANDING

Background

Bike sharing systems are a means of renting bicycles where the process of obtaining membership, rental, and bike return is automated via a network of kiosk locations throughout a city. Using these systems, people are able rent a bike from a one location and return it to a different place on an as-needed basis. Currently, there are over 500 bike-sharing programs around the world.

The data generated by these systems makes them attractive for researchers because the duration of travel, departure location, arrival location, and time elapsed is explicitly recorded. Bike sharing systems therefore function as a sensor network, which can be used for studying mobility in a city.

This information would be useful for city planners, investors, and other NGO's who have any interest in fitness or the environment. As well as anyone dealing with traffic congestion issues in the city core. It is a great internal asset to help the company function more efficiently but also a commodity that could be sold for profit to any of the entities highlighted above.

However, Capital Bikeshare has recently confirmed that their monitoring software for their operation in Washington, D.C. has a defect. An audit from their parent company sparked by lower than expected revenue numbers in 2011 and 2012, uncovered that the hourly total ridership data was missing from the data for any day past the 19th of the month.

As such, they hired SM^{2}R^2 Consulting Group to help recover the lost revenue over the past 2 years.

After negotiations, Capital Bikeshare agreed to remit the usual 25% of the revenue earned based on ridership numbers predicted for the missing data by SM^{2}R^2 Consulting Group.

Objective

Capital Bikeshare was able to provide SM^{2}R^2 Consulting Group all of the data for 2011 and 2012 for the hours in the first 19th days of every month. The company was also able to provide all of the hourly information related to weather conditions for the missing data.

The objective is to use the hourly information available along with a variety of models to predict the number of casual and registered riders to derive the total hourly ridership totals.

SM^{2}R^2 Consulting Group must show that the model chosen for the final prediction represents the best option for predicting numbers by comparing the model's Root Mean Squared Error (RMSE) calculation.

Once the ridership totals are predicted, the payment to the parent company will be 25% of the previously agreed to rates of $ 2.25/casual rider and $1.50/registered rider, plus a 2.5% penalty on the total amount.

DATA UNDERSTANDING

The data used in this analysis represents the raw data as provided by Capital Bikeshare (taken from Kaggle). Two databases were provided, the Bike Sharing (Train) dataset has 12 attributes and 10886 records. Eleven of the fields are numeric with the twelfth being a date field. The Missing Bike Sharing (Test) dataset consists of 9 attributes with 6,493 records. The 9 attributes are the same as the Bike Sharing attributes but missing casual, registered and count.

Description of Bike Sharing dataset

Attributes	Data Type / New Type	Description	Range
datetime	date	hourly date + timestamp	2011-01-01 thru 2012-12-19 (only the first 19 days of each month)
season	numeric / factor	1 = Winter, 2 = Spring, 3 = Summer, 4 = Fall	1,2,3,4
holiday	numeric / factor	whether the day is considered a holiday	0,1
workingday	Numeric / factor	whether the day is neither a weekend nor holiday	0,1
weather	numeric/ factor	1: Clear, Few clouds, Partly cloudy, Partly cloudy (Good)	1,2,3,4
		2: Mist + Cloudy, Mist + Broken clouds, Mist + Few clouds, Mist (Normal)	
		3: Light Snow, Light Rain + Thunderstorm + Scattered clouds, Light Rain +Scattered clouds (Bad)	
		4: Heavy Rain + Ice Pellets + Thunderstorm + Mist, Snow + Fog (Very Bad)	
temp	numeric	temperature in Celsius	0.82 to 41.0 mean =20.23 SD =7.8
atemp	numeric	feels like temperature in Celsius	0.76 to 45.455 Mean = 23.65 SD = 8.47
humidity	numeric	relative humidity	0 to 100 Mean = 61.88 SD = 19.24
windspeed	numeric	wind speed	0 to 57 Mean = 12.8 SD = 8.16
casual	numeric	number of non-registered user rentals initiated	0 to 367 Mean = 36.02 SD = 49.96
registered	numeric	number of registered user rentals initiated	0 to 886 Mean = 155.55 SD = 151.04
count	numeric	number of total rentals both casual and registered	1 to 997 Mean = 191.57 SD = 181.14

The following features are newly created variables which will be used to enhance the model, and/or visualizations for this project

Attributes	Data Type	Description	Range
hour	integer/factor	Hour of the day the ride was booked	0 to 23
month	numeric/factor	Month of the year the ride was booked	1 to 12
year	numeric/factor	Year the ride was booked	2011, 2012
wday	numeric/factor	Day of the week the ride was booked	1 to 7
Temp_Group	Factor	Derived from temperature. Binned into 4 groups based on model recommendations.	Cold, Cool, Warm, Hot
Atemp_Group	Factor	Derived from Humidity. Binned into 4 groups based on business definition recommendations.	Cold, Cool, Warm, Hot
Humidity_Group	Factor	Derived from Humidity. Binned into 3 groups based on model recommendations.	Low, Medium, High
shareHourFact	Factor	Derived from hour.	0 to 23

Visualizations

- Show registered users generally using the bike rental service during rush hour periods
- Casual users tend to use the service more often on the weekend and during mid-day
- The season does not significantly impact the users, but riders were lower in the Winter than the other 3 seasons. Fall, Spring and Summer had very similar ridership levels
- The days of extreme weather were minimal in Washington with only 1 day of being recorded. As expected, ridership is highest when the weather is good followed by normal weather.
- Riders prefer moderate temperatures rather than extremely hot or extremely cold days

Assumptions

The following assumptions were made when modelling the data:

- There were no limitations on bicycle availability as this would impact the potential number of rentals in each period.
- The weather data was accurate and not missing as several data points for wind were zero.

DATA PREPARATION

To perform the analysis certain R libraries were used. The code below was used to load and initialize the libraries.

```
library(library_name)
setwd("C:\\....")
Bikedf=read.csv(file.choose(), header = TRUE)
Bikedf_test = read.csv(file.choose(), header = TRUE)
```

Preview of the data

```
head(Bikedf,n=3)
```

```
             datetime season holiday workingday weather temp  atemp humidity
1 2011-01-01 00:00:00      1       0          0       1 9.84 14.395       81
2 2011-01-01 01:00:00      1       0          0       1 9.02 13.635       80
3 2011-01-01 02:00:00      1       0          0       1 9.02 13.635       80
  windspeed casual registered count
1         0      3         13    16
2         0      8         32    40
3         0      5         27    32
```

Dimension of the dataset

The code below shows the dimension of the dataset.

```
dim(Bikedf)
[1] 10886    12
```

Data attributes summary

This is a quick view of the data attributes statistics. This table shows the min, max, mean and the 1st and 3rd quartile values of the numeric features.

```
summary(Bikedf)
```

```
               datetime         season          holiday            workingday
 2011-01-01 00:00:00:    1   Min.   :1.000   Min.   :0.00000   Min.   :0.0000
 2011-01-01 01:00:00:    1   1st Qu.:2.000   1st Qu.:0.00000   1st Qu.:0.0000
 2011-01-01 02:00:00:    1   Median :3.000   Median :0.00000   Median :1.0000
 2011-01-01 03:00:00:    1   Mean   :2.507   Mean   :0.02857   Mean   :0.6809
 2011-01-01 04:00:00:    1   3rd Qu.:4.000   3rd Qu.:0.00000   3rd Qu.:1.0000
 2011-01-01 05:00:00:    1   Max.   :4.000   Max.   :1.00000   Max.   :1.0000
 (Other)             :10880

    weather          temp            atemp           humidity
 Min.   :1.000   Min.   : 0.82   Min.   : 0.76   Min.   :  0.00
 1st Qu.:1.000   1st Qu.:13.94   1st Qu.:16.66   1st Qu.: 47.00
 Median :1.000   Median :20.50   Median :24.24   Median : 62.00
 Mean   :1.418   Mean   :20.23   Mean   :23.66   Mean   : 61.89
 3rd Qu.:2.000   3rd Qu.:26.24   3rd Qu.:31.06   3rd Qu.: 77.00
 Max.   :4.000   Max.   :41.00   Max.   :45.45   Max.   :100.00

   windspeed          casual          registered        count
 Min.   : 0.000   Min.   :  0.00   Min.   :  0.0   Min.   :  1.0
 1st Qu.: 7.002   1st Qu.:  4.00   1st Qu.: 36.0   1st Qu.: 42.0
```

```
Median :12.998   Median : 17.00   Median :118.0   Median :145.0
Mean   :12.799   Mean   : 36.02   Mean   :155.6   Mean   :191.6
3rd Qu.:16.998   3rd Qu.: 49.00   3rd Qu.:222.0   3rd Qu.:284.0
Max.   :56.997   Max.   :367.00   Max.   :886.0   Max.   :977.0
```

Checking for missing values

any(is.na(Bikedf))

[1] FALSE

The data set has no missing values. The code below calculates the number of rows with missing values.

Structure of the dataset

This code below shows the structure of the dataset.

str(Bikedf)

```
'data.frame':  10886 obs. of  12 variables:
 $ datetime  : Factor w/ 10886 levels "2011-01-01 00:00:00",..: 1 2 3 4 5 6..
 $ season    : int  1 1 1 1 1 1 1 1 1 1 ...
 $ holiday   : int  0 0 0 0 0 0 0 0 0 0 ...
 $ workingday: int  0 0 0 0 0 0 0 0 0 0 ...
 $ weather   : int  1 1 1 1 1 2 1 1 1 1 ...
 $ temp      : num  9.84 9.02 9.02 9.84 9.84 ...
 $ atemp     : num  14.4 13.6 13.6 14.4 14.4 ...
 $ humidity  : int  81 80 80 75 75 75 80 86 75 76 ...
 $ windspeed : num  0 0 0 0 0 ...
 $ casual    : int  3 8 5 3 0 0 2 1 1 8 ...
 $ registered: int  13 32 27 10 1 1 0 2 7 6 ...
 $ count     : int  16 40 32 13 1 1 2 3 8 14 ...
```

Converting the data

To properly assess the data set season, holiday, weather, and workingday should be factors not integers. Datetime should be year, month, day, hour in separate columns. Also, the revaluing of the data points to make them easier to understand is done in this section. The codes below shows the conversion of the data.

```
>Bikedf$season <- as.factor(Bikedf$season)
>Bikedf$holiday <- as.factor(Bikedf$holiday)
>Bikedf$workingday <- as.factor(Bikedf$workingday)
>Bikedf$weather <- as.factor(Bikedf$weather)
Bikedf$season <- revalue(Bikedf$season,
                    c("1"="Winter","2"="Spring","3"="Summer","4"="Fall"))
Bikedf$holiday <- revalue(Bikedf$holiday,
                     c("1"="Holiday","0"="Regular_day"))
Bikedf$workingday <- revalue(Bikedf$workingday,
                        c("1"="Workingday","0"="Weekend/holiday"))
Bikedf$weather <- revalue(Bikedf$weather,
                     c("1"="Good","2"="Normal","3"="Bad","4"="Very Bad"))
>Bikedf %>%
mutate(datetime = fastPOSIXct(datetime, "GMT")) %>%
  mutate(hour = hour(datetime),
```

```
month = month(datetime),
year = year(datetime),
wday = wday(datetime)) -> Bikedf
```

Bin the temp, atemp, humidity and hour variables for business use and graphing purposes. We converted the hour feature to factor for better modelling results

```
Bikedf <- Bikedf%>%
  mutate (
    Temp_Group = cut(Bikedf$temp, breaks = c(-40, 12.71, 22.55, 29.93, 50) , labels = c("Cold
<12.71","Cool 12.71 - 22.55","Warm 22.55 - 29.9", "Hot <29.93")),
    Atemp_Group = cut(Bikedf$atemp,4, labels = c("Cold","Cool","Warm", "Hot")),
    Humidity_Group = cut(Bikedf$humidity,breaks = c(-1, 46.5, 66.5, 101), labels =
c("Low","Medium","High")),
    ShareHourFact = as.factor(hour(datetime))
  )
```

Used the rpart function to determine bin breakpoints

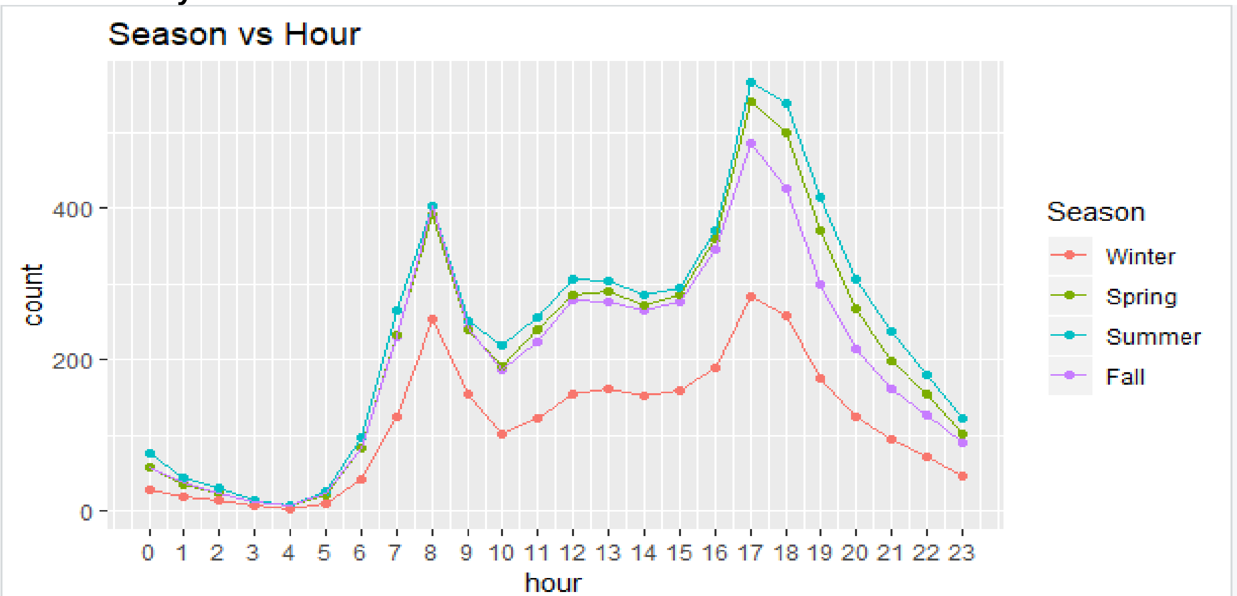
```
temp1 <- rpart(Bikedf$casual ~ Bikedf$temp)
plot(temp1)
text(temp1)

humidity1 <- rpart(Bikedf$count ~ Bikedf$humidity)
plot(humidity1)
text(humidity1)

atemp1 <- rpart(Bikedf$registered ~ Bikedf$atemp)
plot(atemp1)
text(atemp1)
```

Now let's take a closer look at the relationship between bikes rented by hour vs. season, holiday, workingday, and weather. (see graphs below)

```
>ggplot(Bikedf, aes(x=hour, y=count, color=season))+geom_point(data = season_vs_hour,
aes(group = season))+geom_line(data = season_vs_hour, aes(group =
season))+ggtitle("Season vs Hour")+ scale_colour_hue('Season',breaks =
levels(Bikedf$season))+ scale_x_continuous(breaks = seq(0, 23, by = 1))
```

The same code was used to plot the weather, holiday and working day graphs.

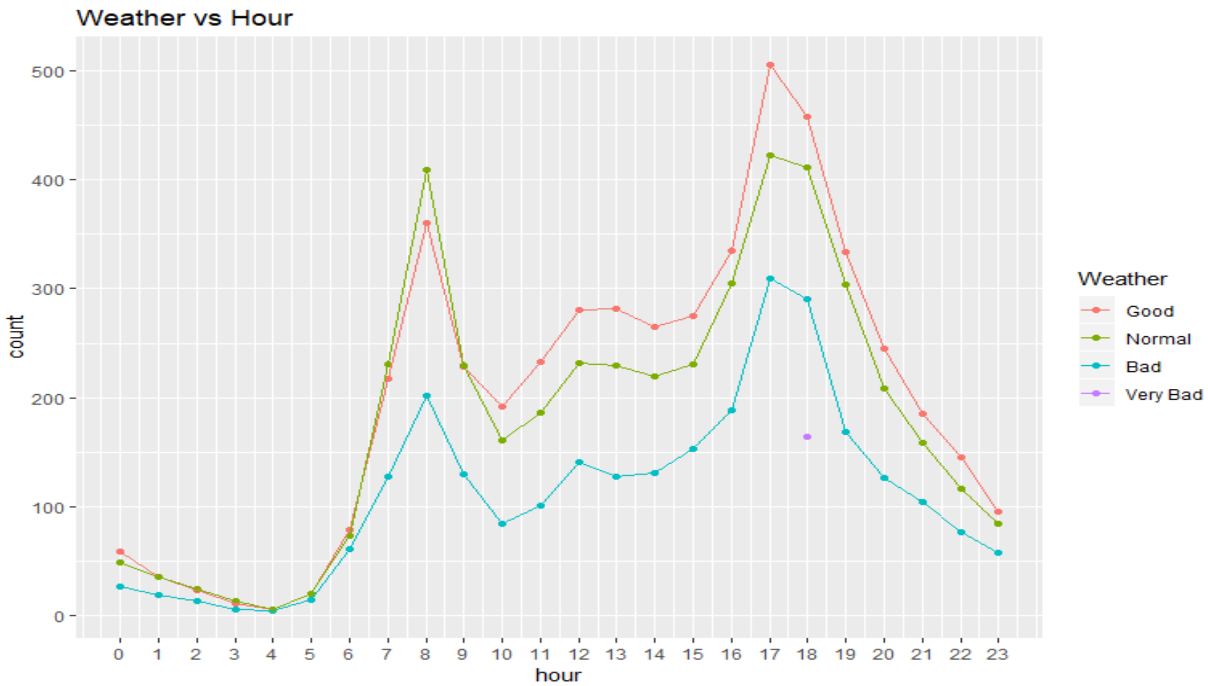
Bike rental by season



This graph shows there are more bike rentals in the morning from 7-9 and in the evening from 16-19 hours and people rent bikes more in Spring and Summer, and much less in Winter.

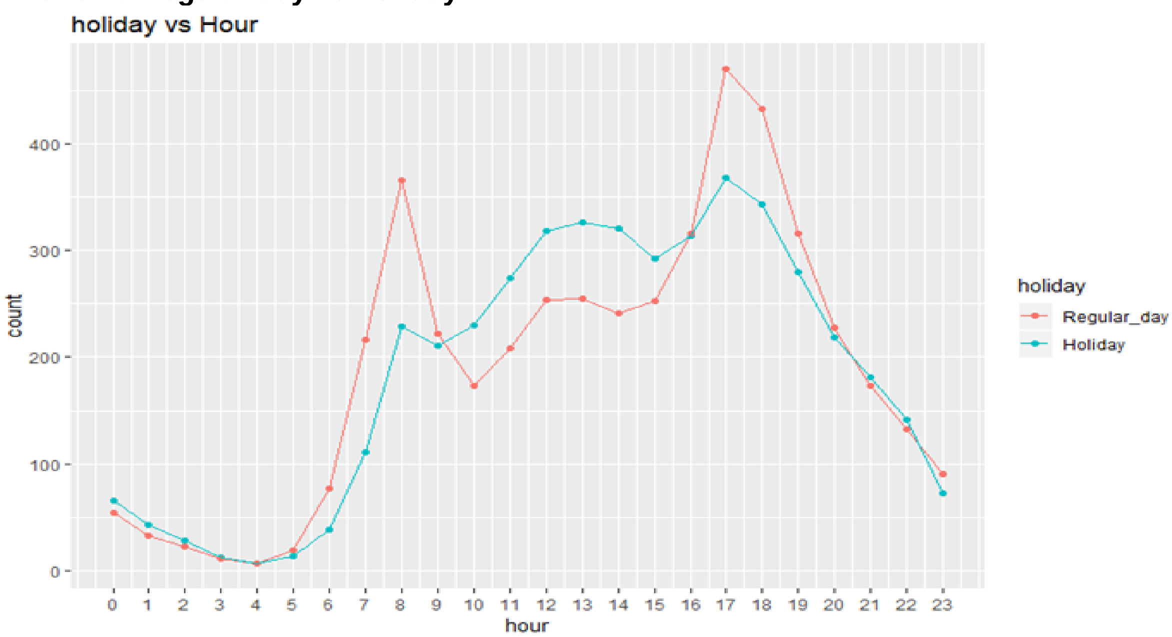
ggplot(data=Bikedf) + geom_bar(mapping = aes(x = Temp_Group, fill = workingday))

The most popular temperature to rent is between 12.7 and 22.5 degrees. If it gets warmer then ridership begins to decline.



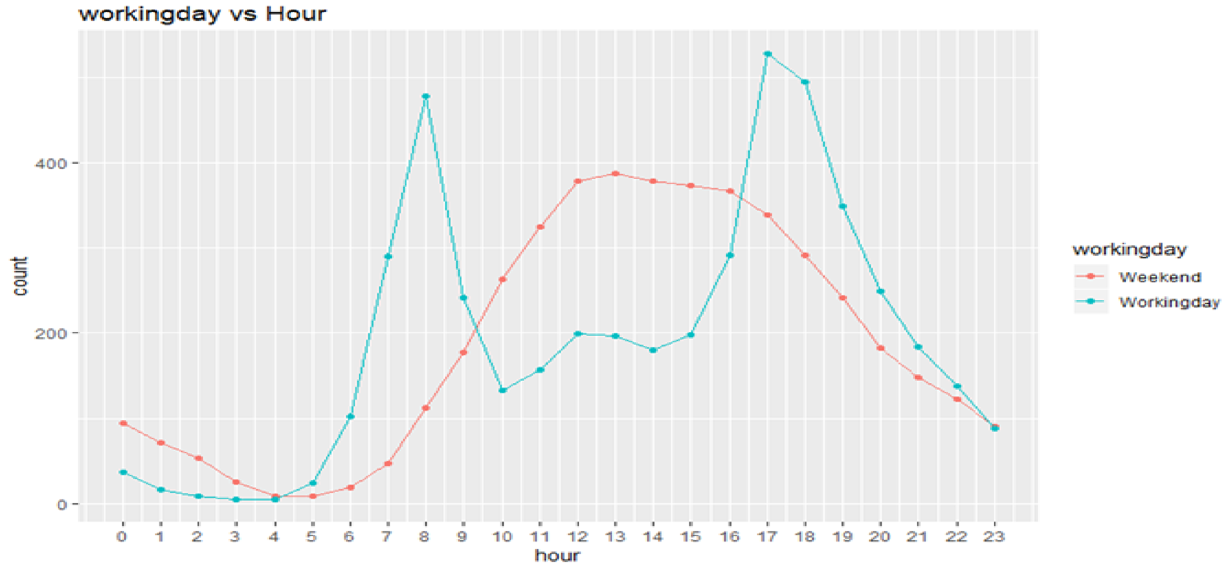
This plot shows riders rent bikes more bikes when the weather is Good; generally, in Very Bad weather conditions no bikes are being rented.

Bike rental regular day vs holiday



This graph shows there are more bikes rented on a regular day between 6:00-9:00 and 16:00-20:00. This would most likely be because riders are traveling to and from work between these times on working days. On holidays riders rent more bikes between 9:00-18:00 hour more than during a regular day.

Working day vs weekend

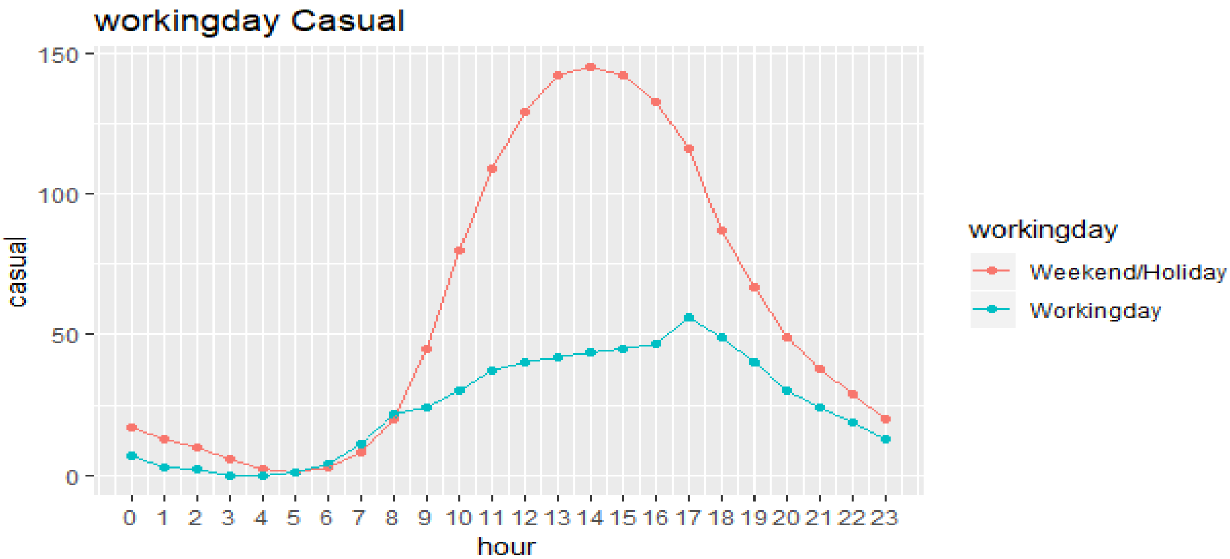


This graph shows more bikes rented on weekends during the hours of 9:00-16:00 than on a working day.

Comparing the rental records for casual **and** registered **riders**

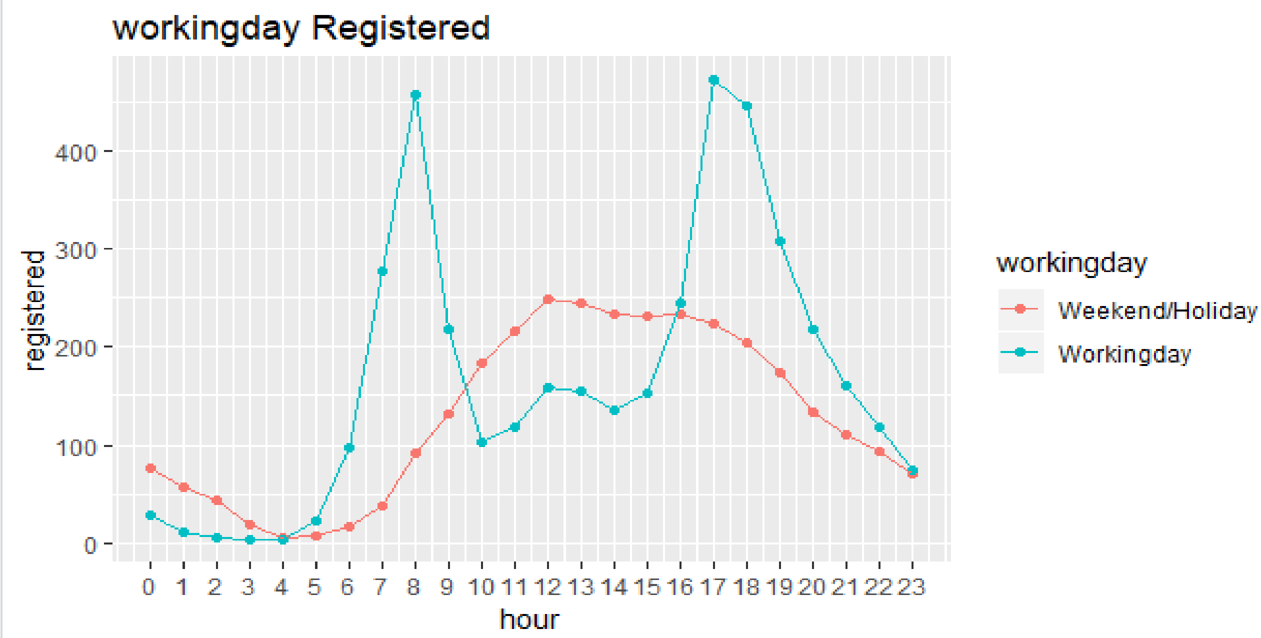
>workingday_casual <- sqldf('select workingday, hour, avg(casual) as casual from Bikedf group by workingday, hour')
>workingday_registered <- sqldf('select workingday, hour, avg(registered) as registered from Bikedf group by workingday, hour')

>ggplot(Bikedf, aes(x=hour, y=casual, color=workingday))+geom_point(data = workingday_casual, aes(group = workingday))+geom_line(data = workingday_casual, aes(group = workingday))+ggtitle("workingday Casual")+ scale_colour_hue('workingday',breaks = levels(Bikedf$workingday))+ scale_x_continuous(breaks = seq(0, 23, by = 1))



>ggplot(Bikedf, aes(x=hour, y=registered, color=workingday))+geom_point(data = workingday_registered, aes(group = workingday))+geom_line(data = workingday_registered, aes(group =

workingday))+ggtitle("workingday Registered")+ scale_colour_hue('workingday',breaks = levels(Bikedf$workingday))+ scale_x_continuous(breaks = seq(0, 23, by = 1))



From the graphs above we can see the registered customer and casual customer have different rental behaviors.

- Casual customers prefer renting bikes on weekends during the daytime.
- Registered customers prefer renting bikes on working days in the morning between 7:00-9:00 and evening from 16:00-19:00 hour. They are most likely to be people travelling to work.
- Which also tells us, we need to predict the number of casual and registered customers separately.

Summary of temp, atemp, humidity and windspeed

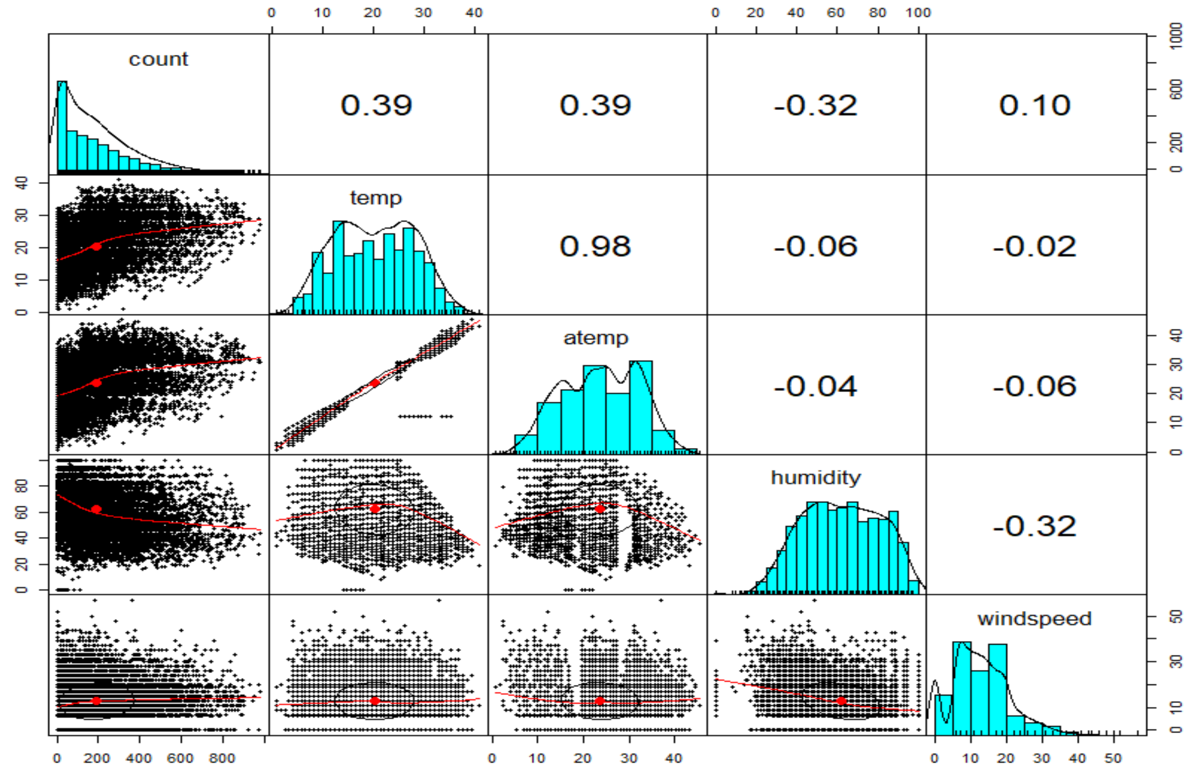
```
>summary(Bikedf$temp)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   0.82   13.94   20.50   20.23   26.24   41.00
>summary(Bikedf$atemp)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   0.76   16.66   24.24   23.66   31.06   45.45
>summary(Bikedf$humidity)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   0.00   47.00   62.00   61.89   77.00  100.00
>summary(Bikedf$windspeed)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  0.000   7.002  12.998  12.799  16.998  56.997
```

Finding the correlation

Let’s take the rest of variables into consideration: temp, atemp, humidity, and windspeed. The codes below show us a correlation plot.

Is there any correlation between the count of Bikes rented and the above attributes?
Casual, registered and count show similar results in the correlation plot.

>pairs.panels(Bikedf[c("count","temp","atemp","humidity","windspeed")],gap=0)



temp: normal distribution the highest count is around 20 degrees
atemp: strong correlation with temp
humidity: close to a normal distribution
windspeed: very right-skewed, distributions that are skewed right have a tail

Independent and Dependent Variables

The attributes temp, atemp, humidity and windspeed are considered dependent variables (DV's); count, casual and registered are the independent variables (IV's). Our assumption is that there is that a relationship exists between these 4 DV's and the IV's.

Now how do we know that this assumption is validated in our data? The codes below for casual and registered customers will help to answer this question.

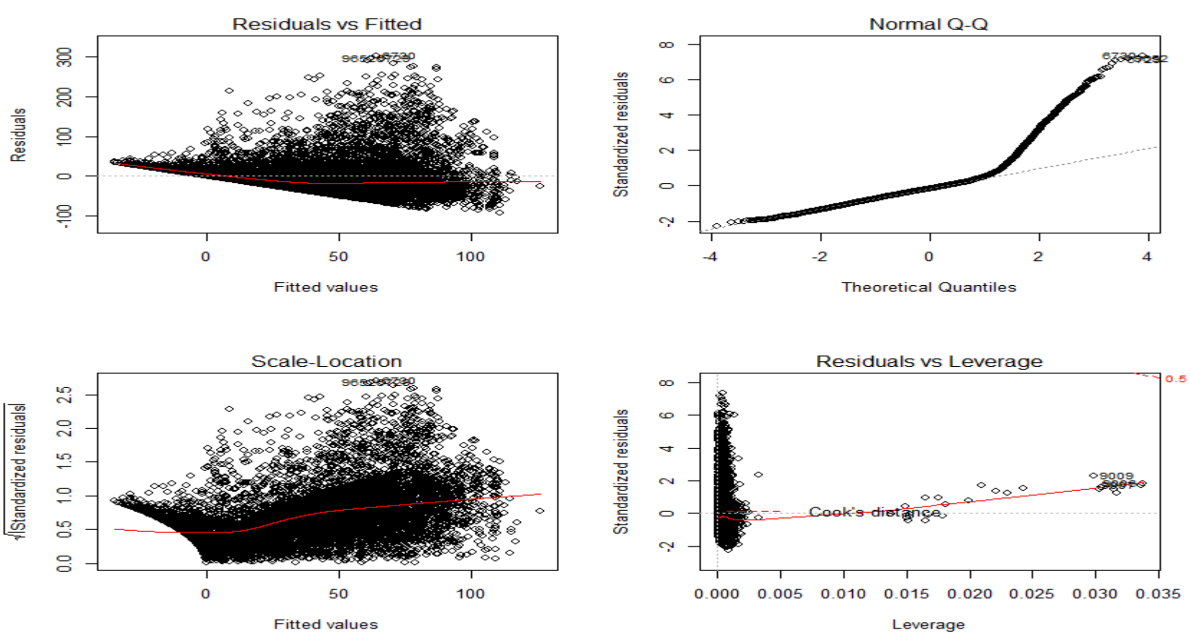
Casual Customer:

>Bikedf.casual.fit <- lm(Bikedf$casual ~ Bikedf$temp + Bikedf$atemp + Bikedf$humidity + Bikedf$windspeed,data=Bikedf)
>summary(Bikedf.casual.fit)

```
Call:
```

```
lm(formula = Bikedf$casual ~ Bikedf$temp + Bikedf$atemp + Bikedf$humidity +
    Bikedf$windspeed, data = Bikedf)

Residuals:
    Min      1Q  Median      3Q     Max
-93.320 -22.054  -6.701   9.353 302.870

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)      25.64862    2.15475  11.903  < 2e-16 ***
Bikedf$temp       0.96655    0.30101   3.211  0.00133 **
Bikedf$atemp      1.76927    0.27698   6.388 1.75e-10 ***
Bikedf$humidity  -0.83668    0.02170 -38.554  < 2e-16 ***
Bikedf$windspeed  0.05832    0.05210   1.119  0.26303
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 41.14 on 10881 degrees of freedom
Multiple R-squared:  0.3222,  Adjusted R-squared:  0.3219
F-statistic:  1293 on 4 and 10881 DF,  p-value: < 2.2e-16
```

> plot(Bikedf.casual.fit)

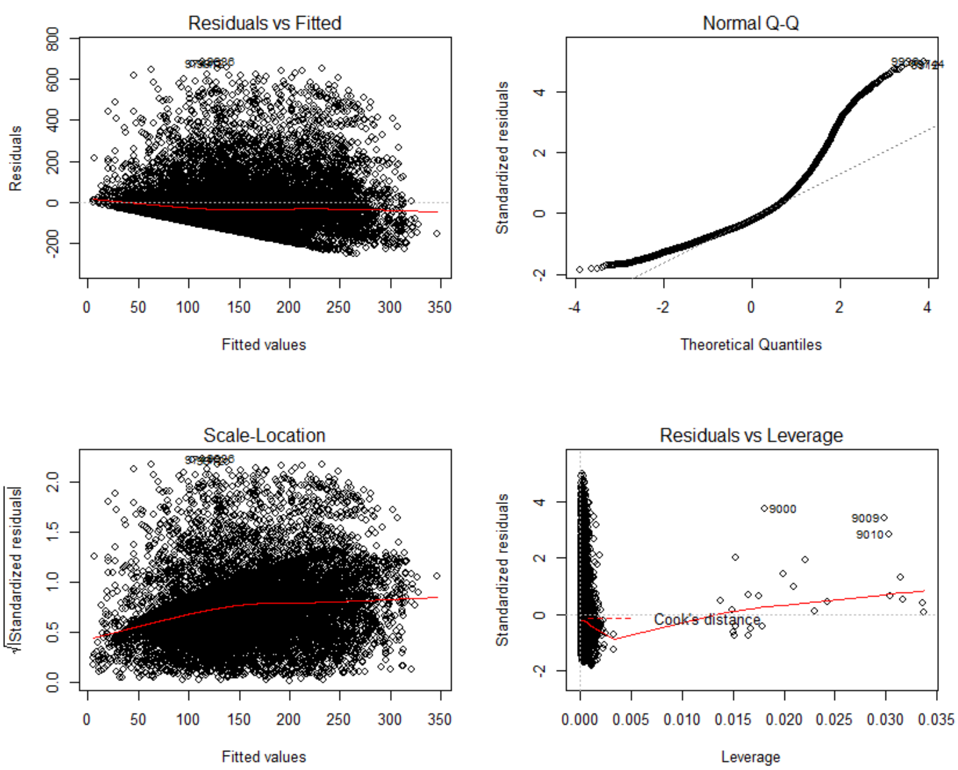


Registered Customer:

>Bikedf.registered.fit <- lm(Bikedf$registered ~ Bikedf$temp + Bikedf$atemp + Bikedf$humidity + Bikedf$windspeed,data=Bikedf)
>summary(Bikedf.registered.fit)

```
Call:
lm(formula = Bikedf$registered ~ Bikedf$temp + Bikedf$atemp +
    Bikedf$humidity + Bikedf$windspeed, data = Bikedf)

Residuals:
    Min      1Q  Median      3Q     Max
-254.45  -87.73  -33.63   48.13  689.13

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)      138.72449    7.23642  19.170  < 2e-16 ***
Bikedf$temp        1.44573    1.01090   1.430  0.15270
Bikedf$atemp       4.14072    0.93021   4.451 8.62e-06 ***
Bikedf$humidity   -1.89382    0.07288 -25.985  < 2e-16 ***
Bikedf$windspeed   0.53378    0.17497   3.051  0.00229 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 138.2 on 10881 degrees of freedom
Multiple R-squared:  0.1635,   Adjusted R-squared:  0.1632
F-statistic: 531.9 on 4 and 10881 DF,  p-value: < 2.2e-16
```

> plot((Bikedf.registered.fit))



In the analysis for the graphs above, both casual and registered data show similar results.

Residential vs. Fitted:
Ideally, this plot shouldn't show any pattern. Seeing almost equally spread residuals on either side of a horizontal line without distinct patterns is a good indication there are no non-linear relationships. It seems like both graphs are close to a funnel shape pattern, which indicates the data might suffer heteroskedasticity.

Normal Q-Q:
Ideally, the plot should be showing a straight line, but it does show a curved line. Which means it has a non-normal distribution.

Scale-Location:
Ideally, the plot shouldn't show any patterns, but both graphs show slight heteroskedasticity.

Residential vs. Leverage:
Based on both graphs, even though the data have extreme values, they would not be influential to determine a regression line. That means, the results wouldn't be much different if we either include or exclude them from the analysis

Assumption Violation

Let's use some quick methods to check the assumption violations again:

- **Durbin Watson Statistic (DW)**

```
> dwtest(Bikedf.casual.fit)
        Durbin-Watson test
data:  Bikedf.casual.fit
DW = 0.17099, p-value < 2.2e-16
alternative hypothesis: true autocorrelation is greater than 0
> dwtest(Bikedf.registered.fit)
        Durbin-Watson test
data:  Bikedf.registered.fit
DW = 0.48031, p-value < 2.2e-16
alternative hypothesis: true autocorrelation is greater than 0
```

  Both 0<DW<2 means positive autocorrelation.

- **Variance Inflation Factor (VIF**)

```
> vif(Bikedf.casual.fit)
   Bikedf$temp    Bikedf$atemp  Bikedf$humidity Bikedf$windspeed
     35.376203       35.436388         1.121811         1.163748
> vif(Bikedf.registered.fit)
   Bikedf$temp    Bikedf$atemp  Bikedf$humidity Bikedf$windspeed
     35.376203       35.436388         1.121811         1.163748
```

  Both VIF > 10: high multicollinearity, temp & atemp are the cause of this.

- **Breusch-Pagan/Cook Weisberg Test**

```
> ncvTest(Bikedf.casual.fit)
Non-constant Variance Score Test
Variance formula: ~ fitted.values
Chisquare = 3889.262, Df = 1, p = < 2.22e-16

> ncvTest(Bikedf.registered.fit)
Non-constant Variance Score Test
Variance formula: ~ fitted.values
Chisquare = 347.112, Df = 1, p = < 2.22e-16
```

  P value <0.05: the heteroskedasticity is present

Determining the Importance of Features using Random Forest Feature Plot:

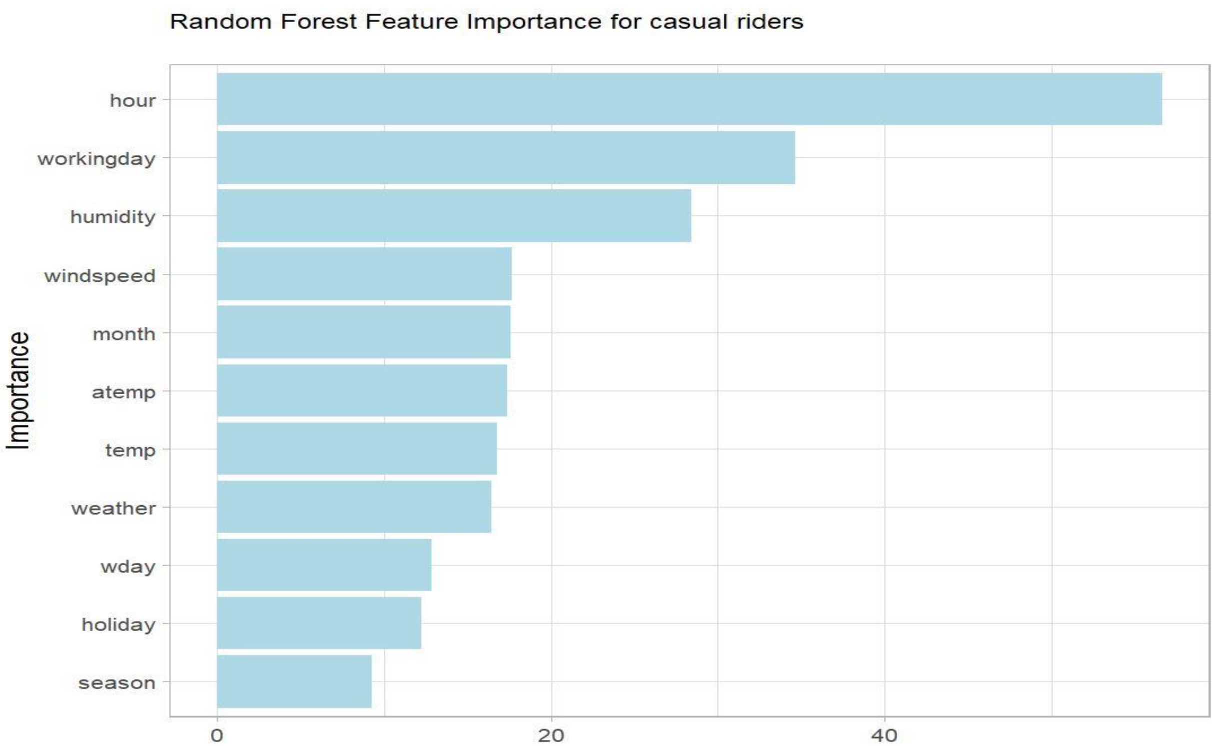
The Extract Features function will be used for both codes below:

```
>extractFeatures <- function(data) {features<-
c("season","holiday","workingday","weather","temp","atemp","humidity","windspeed","hour","wday","month")
```

```
return(data[,features])}

>set.seed(123)
>rfcas <- randomForest(extractFeatures(Bikedf), bike$casual, ntree=100, importance=TRUE)
>impcas <- importance(rfcas, type=1)
>featureImportancecas <- data.frame(Feature=row.names(impcas), Importance=impcas[,1])
```

Plotting the variable importance – casual riders

```
>BikedfFIcasual <- ggplot(featureImportancecas, aes(x=reorder(Feature, Importance), y=Importance)) +
geom_bar(stat="identity", fill="lightblue") + coord_flip() + theme_light(base_size=20) + xlab("Importance") +
ylab("") + ggtitle("Random Forest Feature Importance for casual riders\n") +
theme(plot.title=element_text(size=18))
>BikedfFIcasual
```

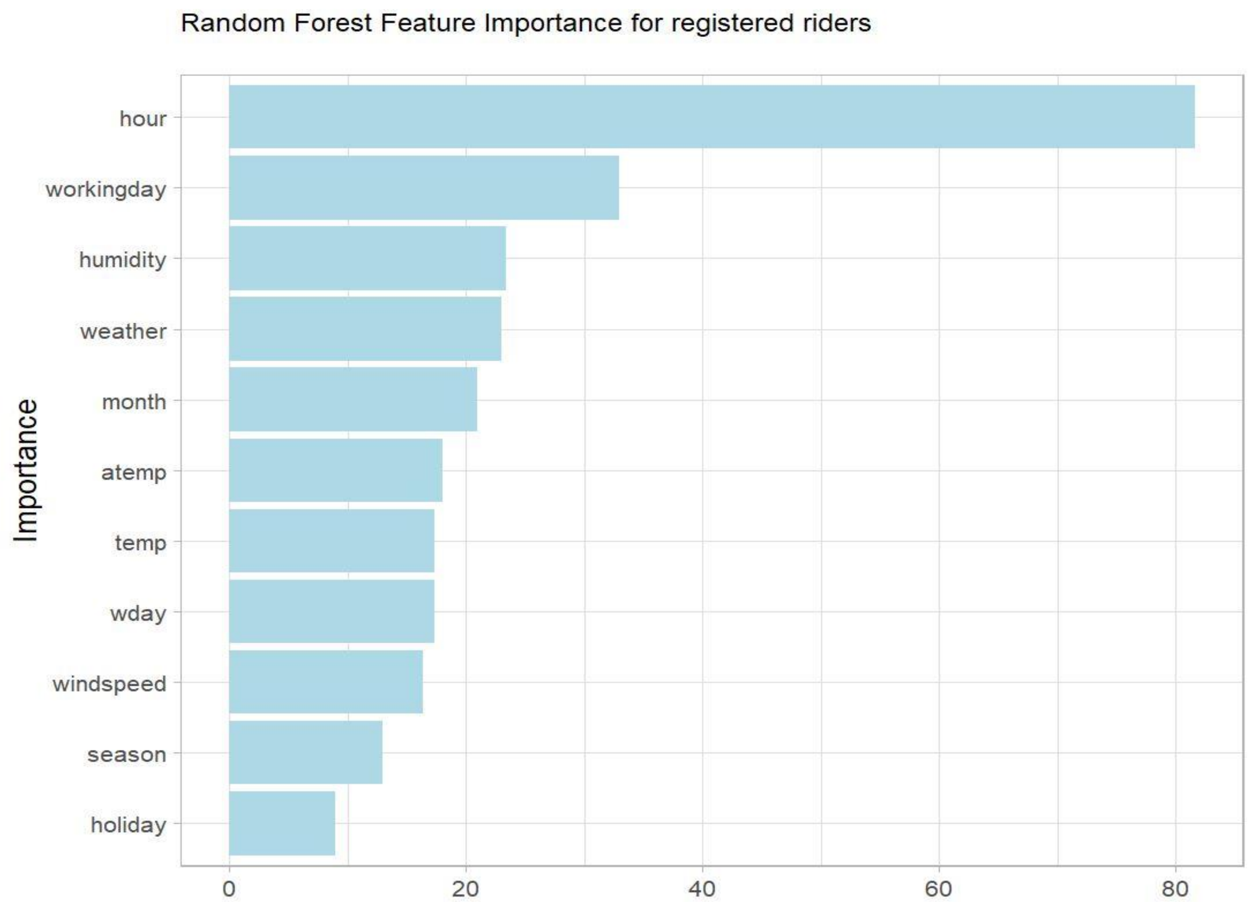


Here is the code for the variable importance for registered riders:

```
>set.seed(123)
>rfreg <- randomForest(extractFeatures(Bikedf), Bikedf$registered, ntree=100, importance=TRUE)
>impreg <- importance(rfreg, type=1)
>featureImportancereg <- data.frame(Feature=row.names(impreg), Importance=impreg[,1])
```

Plotting the variable importance – registered riders

```
>BikedfFIreg <- ggplot(featureImportancereg, aes(x=reorder(Feature, Importance), y=Importance)) +
geom_bar(stat="identity", fill="lightblue") + coord_flip() + theme_light(base_size=20) + xlab("Importance") +
ylab("") + ggtitle("Random Forest Feature Importance for registered riders\n") +
theme(plot.title=element_text(size=18))
>BikedfFIreg
```



There are a number of differences in the importance of the dependent variables between the casual & registered riders.

- The strength of the correlation eg. hour for registered riders is >80 while for casual riders it is < 60
- The order of the DV's is different:

Casual	**Registered**
hour	hour
workingday	workingday
humidity	humidity
windspeed	weather
month	month
atemp	atemp
temp	temp
weather	wday
wday	windspeed
holiday	season
season	holiday

We decided to choose workingday**+**weather**+**humidity**+**hour**+**wday**+**month **as the variables for our model.**

Creating new dataset for Modeling purpose
Bikedf1<-Bikedf

Converting the variable to factors for better accuracy
Bikedf1$month = as.factor(Bikedf1$month)
Bikedf1$wday = as.factor(Bikedf1$wday)
Bikedf1$hour = as.factor(Bikedf1$hour)
Bikedf1$season = as.factor(Bikedf1$season)
Bikedf1$holiday = as.factor(Bikedf1$holiday)
Bikedf1$workingday = as.factor(Bikedf1$workingday)
Bikedf1$weather = as.factor(Bikedf1$weather)

MODELING

We chose 4 different models to find the most effective one; GLM, Random forest, XGBLinear and XGBTree. After examining the data, we decided to use the log function + 1 with all of the models to handle the data points where there were zero riders.

1. XGBTREE

```
#Registered Riders
set.seed(123)
# define training control & train model
train_control <- trainControl(method="cv", number=5)
# train the model
model <- train(log(registered+1)~workingday+weather+humidity+hour+wday+month,
data=Bikedf1, trControl=train_control, method="xgbTree")
# summarize results
print(model)
mean(model$results$Rsquared)
[1] 0.8018961

mean(model$results$RsquaredSD)
[1] 0.006848086

XGBTree_RMSE_reg = mean(model$results$RMSE)
[1] 0.6219303

#Casual Riders
set.seed(123)
# define training control & train model
train_control2 <- trainControl(method="cv", number=5)
model2 <- train(log(casual+1)~atemp+humidity+windspeed+hour+workingday+month,
data=Bikedf1, trControl=train_control2, method="xgbTree")
# summarize results
print(model2)
mean(model2$results$Rsquared)
[1] 0.8128748

mean(model2$results$RsquaredSD)
[1] 0.0062544

XGBTree_RMSE_cas = mean(model2$results$RMSE)
[1] 0.6459077
```

2. XGBLINEAR

```
#Registered Riders
set.seed(123)
# define training control & train model
train_control3 <- trainControl(method="cv", number=5)
model3 <- train(log(registered+1)~workingday+weather+humidity+hour+wday+month,
```

```
              data=Bikedf1, trControl=train_control3, method="xgbLinear")
# summarize results
print(model3)
mean(model3 $results$Rsquared)
[1] 0.879045

mean(model3$results$RsquaredSD)
[1] 0.005582614

XGBLinear_RMSE_reg = mean(model3$results$RMSE)
[1] 0.486871
```

#Casual Riders

```
set.seed(123)
# define training control
train_control4 <- trainControl(method="cv", number=5)
model4 <- train(log(casual+1)~atemp+humidity+windspeed+hour+workingday+month,
              data=Bikedf1, trControl=train_control4, method="xgbLinear")
# summarize results
print(model4)
mean(model4$results$Rsquared)
[1] 0.8427766

mean(model4$results$RsquaredSD)
[1] 0.003902955

XGBLinear_RMSE_cas = mean(model4$results$RMSE)
[1] 0.5922195
```

3. RANDOM FOREST REGRESSION

#Registered Riders

```
set.seed(123)
# define training control & train model
train_control5 <- trainControl(method="cv", number=5)
# train the model
model5 <- train(log(registered+1)~workingday+weather+humidity+hour+wday+month,
              data=Bikedf1, trControl=train_control5, method="ranger")
# summarize results
print(model5)
mean(model5$results$Rsquared)
[1] 0.8479719
mean(model5$results$RsquaredSD)
[1] 0.005923433
RandomForest_RMSE_reg = mean(model5$results$RMSE)
[1] 0.6354266
```

#Casual Riders

```
set.seed(123)
# define training control & train model
train_control6 <- trainControl(method="cv", number=5)
```

```
model6 <- train(log(casual+1)~atemp+humidity+windspeed+hour+workingday+month,
        data=Bikedf1, trControl=train_control6, method="ranger")
# summarize results
print(model6)
mean(model6$results$Rsquared)
[1] 0.827893
mean(model6$results$RsquaredSD)
[1] 0.006953266
RandomForest_RMSE_cas = mean(model6$results$RMSE)
[1] 0.6986834
```

4. GENERALIZED LINEAR REGRESSION

```
#Registered riders
set.seed(123)
# define training control & train model
train_control7 <- trainControl(method="cv", number=5)
model7 <- train(log(registered+1)~workingday+weather+humidity+hour+wday+month,
        data=Bikedf1, trControl=train_control7, method="glm")
# summarize results
print(model7)
Generalized Linear Model

10886 samples
    6 predictor

No pre-processing
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 8709, 8709, 8709, 8709, 8708
Resampling results:

  RMSE       Rsquared   MAE
  0.6517985  0.7831748  0.4977766

GeneralLinearRegression_RMSE_reg = mean(model7$results$RMSE)
[1] 0.6517985

#Casual Riders
set.seed(123)
# define training control & train model
train_control8 <- trainControl(method="cv", number=5)
# train the model
model8 <- train(log(casual+1)~atemp+humidity+windspeed+hour+workingday+month,
        data=Bikedf1, trControl=train_control8, method="glm")
# summarize results
print(model8)
Generalized Linear Model

10886 samples
    6 predictor

No pre-processing
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 8709, 8710, 8708, 8708, 8709
Resampling results:
```

```
  RMSE       Rsquared   MAE      
  0.6517697  0.8091515  0.5034748
```

GeneralLinearRegression_RMSE_cas = mean(model8$results$RMSE)

```
[1] 0.6517697
```

Summary of the RMSE for models run:

#registered:

Reg_RMSE = cbind(XGBTree_RMSE_reg, XGBLinear_RMSE_reg,
RandomForest_RMSE_reg, GeneralLinearRegression_RMSE_reg)
Reg_RMSE = as.data.frame(Reg_RMSE)
names(Reg_RMSE) = c("XGBTree_RMSE_reg", "XGBLinear_RMSE_reg",
"RandomForest_RMSE_reg", "GeneralLinearRegression_RMSE_reg")
Reg_RMSE= melt(Reg_RMSE, measure.vars=c("XGBTree_RMSE_reg",
"XGBLinear_RMSE_reg",
RandomForest_RMSE_reg", "GeneralLinearRegression_RMSE_reg"))
Reg_RMSE

```
                          variable     value
1                 XGBTree_RMSE_reg 0.6219303
2               XGBLinear_RMSE_reg 0.4868710
3            RandomForest_RMSE_reg 0.6354266
4 GeneralLinearRegression_RMSE_reg 0.6517985
```

Reg_RMSE[which.min(Reg_RMSE$value),]

```
            variable    value
2 XGBLinear_RMSE_reg 0.486871
```

#Casual:

Cas_RMSE = cbind(XGBTree_RMSE_cas, XGBLinear_RMSE_cas,
RandomForest_RMSE_cas, GeneralLinearRegression_RMSE_cas)
Cas_RMSE = as.data.frame(Cas_RMSE)
names(Cas_RMSE) = c("XGBTree_RMSE_cas", " XGBLinear_RMSE_cas",
"RandomForest_RMSE_cas", "GeneralLinearRegression_RMSE_cas")
Cas_RMSE= melt(Cas_RMSE, measure.vars=c("XGBTree_RMSE_cas",
"XGBLinear_RMSE_cas", "RandomForest_RMSE_cas"))
"GeneralLinearRegression_RMSE_cas"))
Cas_RMSE

```
                          variable     value
1                 XGBTree_RMSE_cas 0.6459077
2               XGBLinear_RMSE_cas 0.5922195
3            RandomForest_RMSE_cas 0.6986834
4 GeneralLinearRegression_RMSE_cas 0.6517697
```

Cas_RMSE[which.min(Cas_RMSE$value),]

```
            variable     value
2 XGBLinear_RMSE_cas 0.5922195
```

Results:
> XGBTree_RMSE_reg
[1] 0.5088946
> XGBTree_RMSE_cas
[1] 0.5161217
> XGBLinear_RMSE_reg
[1] 0.3676079
> XGBLinear_RMSE_cas
[1] 0.454467
RandomForest_RMSE_reg
[1] 0.4786731
RandomForest_RMSE_cas
[1] 0.4488726
> GeneralLinearRegression_RMSE_reg
[1] 0.652325
> GeneralLinearRegression_RMSE_cas
[1] 0.6517697

EVALUATION

Model Evaluation Chart

	Model Name	Customer Type	R2	RMSE Log	RMSE
Model 1	XGBTree	Casual	0.8459105	0.5161217	27.27235
		Registered	0.8501343	0.5088946	57.67994
Model 2	XGBLinear	Casual	0.881446	0.454467	22.89981
		Registered	0.9246329	0.3676079	28.66027
Model 3	RandomForest	Casual	0.8853079	0.4488726	26.29015
		Registered	0.8875713	0.4786731	50.17874
Model 4	GeneralLinear Regression	Casual	0.8091515	0.6517697	32.71331
		Registered	0.7829153	0.652325	71.57386

Base on the above analysis, XGBLinear Model is chosen for Registered Customer and RandomForest is chosen for Casual Customer.

Apply Selected Model to test data:

XGBLINEAR for Registered Customer

```
#Registered Riders
set.seed(123)
xgbTuningGrid = expand.grid(nrounds = 100,
                            lambda = 0.1,
                            alpha = 0.1,
                            eta = 0.3)

# train the model
model_final1 <- train(registered~workingday+weather+humidity+hour+wday+month,
                      data=Bikedf1, tuneGrid = xgbTuningGrid, method="xgbLinear")
# summarize results
print(model_final1)
```

```
eXtreme Gradient Boosting

10886 samples
    6 predictor

No pre-processing
Resampling: Bootstrapped (25 reps)
Summary of sample sizes: 10886, 10886, 10886, 10886, 10886, 10886, ...
Resampling results:

  RMSE       Rsquared   MAE
  0.5421761  0.8670563  0.3953158

Tuning parameter 'nrounds' was held constant at a value of 100
Tuning parameter 'lambda' was held constant at a value of 0.1

Tuning parameter 'alpha' was held constant at a value of 0.1
Tuning parameter 'eta' was held constant at a value of 0.3
```

```
#Import dataset to predict
Bikedf_test = read.csv(file.choose(), header = TRUE)

#make the same changes that were made to Bikedf1.

Bikedf_test$date = as.Date(Bikedf_test$datetime)
Bikedf_test$month = month(Bikedf_test$datetime)
Bikedf_test$wday = wday(Bikedf_test$datetime)
Bikedf_test$hour = hour(Bikedf_test$datetime)

# Reorder columns to keep date fields all together near the beginning.
col_order = c("datetime", "date","month", "wday", "hour", "season", "holiday", "workingday",
"weather", "temp", "atemp", "humidity", "windspeed")
Bikedf_test = Bikedf_test[, col_order]
head(Bikedf_test)
```

```
            datetime       date month wday hour season holiday workingday weather  temp  atemp humidity windspeed
```

```
1 2011-01-20 00:00:00 2011-01-20     1    5    0      1       0          1       1 10.66 11.365       56    26.0027
2 2011-01-20 01:00:00 2011-01-20     1    5    1      1       0          1       1 10.66 13.635       56     0.0000
3 2011-01-20 02:00:00 2011-01-20     1    5    2      1       0          1       1 10.66 13.635       56     0.0000
4 2011-01-20 03:00:00 2011-01-20     1    5    3      1       0          1       1 10.66 12.880       56    11.0014
5 2011-01-20 04:00:00 2011-01-20     1    5    4      1       0          1       1 10.66 12.880       56    11.0014
6 2011-01-20 05:00:00 2011-01-20     1    5    5      1       0          1       1  9.84 11.365       60    15.0013
```

Convert all columns to proper format

Bikedf_test$month = as.factor(Bikedf_test$month)
Bikedf_test$wday = as.factor(Bikedf_test$wday)
Bikedf_test$hour = as.factor(Bikedf_test$hour)
Bikedf_test$season = as.factor(Bikedf_test$season)
Bikedf_test$holiday = as.factor(Bikedf_test$holiday)
Bikedf_test$workingday = as.factor(Bikedf_test$workingday)
Bikedf_test$weather = as.factor(Bikedf_test$weather)

Rename data points to match Bikedf1 data set
Bikedf_test$season <- revalue(Bikedf_test$season,
c("1"="Winter","2"="Spring","3"="Summer","4"="Fall"))
Bikedf_test$holiday <- revalue(Bikedf_test$holiday,
c("1"="Holiday","0"="Regular_day"))
Bikedf_test$workingday <- revalue(Bikedf_test$workingday,
c("1"="Workingday","0"="Weekend/holiday"))
Bikedf_test$weather <- revalue(Bikedf_test$weather,
c("1"="Good","2"="Normal","3"="Bad","4"="Very Bad"))

str(Bikedf_test) – note the changes to the variables from numeric/integer to factor

```
'data.frame':	6493 obs. of  13 variables:
 $ datetime  : Factor w/ 6493 levels "2011-01-20 00:00:00",..: 1 2 3 4 5 6 7 8 9 10 ..
.
 $ date      : Date, format: "2011-01-20" "2011-01-20" "2011-01-20" "2011-01-20" ...
 $ month     : Factor w/ 12 levels "1","2","3","4",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ wday      : Factor w/ 7 levels "1","2","3","4",..: 5 5 5 5 5 5 5 5 5 5 ...
 $ hour      : Factor w/ 24 levels "0","1","2","3",..: 1 2 3 4 5 6 7 8 9 10 ...
 $ season    : Factor w/ 4 levels "1","2","3","4": 1 1 1 1 1 1 1 1 1 1 ...
 $ holiday   : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
 $ workingday: Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 2 ...
 $ weather   : Factor w/ 4 levels "1","2","3","4": 1 1 1 1 1 1 1 1 1 2 ...
 $ temp      : num  10.7 10.7 10.7 10.7 10.7 ...
 $ atemp     : num  11.4 13.6 13.6 12.9 12.9 ...
 $ humidity  : int  56 56 56 56 56 60 60 55 55 52 ...
 $ windspeed : num  26 0 0 11 11 ...
```

Predict values based on model chosen with the parameters from the best result for that model

Prediction = predict(model_final1, Bikedf_test)
Bikedf_test$registered = Prediction
head(Bikedf_test)

```
             datetime       date month wday hour season holiday workingday weather  temp  atemp humidity windspeed registered
1 2011-01-20 00:00:00 2011-01-20     1    5    0      1       0          1       1 10.66 11.365       56   26.0027  56.720328
```

```
2 2011-01-20 01:00:00 2011-01-20      1     5     1      1       0          1       1 10.66 13.635        56     0.0000    7.090939
3 2011-01-20 02:00:00 2011-01-20      1     5     2      1       0          1       1 10.66 13.635        56     0.0000    3.531820
4 2011-01-20 03:00:00 2011-01-20      1     5     3      1       0          1       1 10.66 12.880        56    11.0014    2.527886
5 2011-01-20 04:00:00 2011-01-20      1     5     4      1       0          1       1 10.66 12.880        56    11.0014    2.326857
6 2011-01-20 05:00:00 2011-01-20      1     5     5      1       0          1       1  9.84 11.365        60    15.0013   14.501264
```

summary(Bikedf_test$registered)

```
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max. 
  0.2021  44.3071 111.2404 140.9897 199.9022 832.1545
```

RandomForest for Casual Customers

#Casual

set.seed(123)

train the model

model_final2 <- train(log(casual+.1)~atemp+humidity+windspeed+hour+workingday+month,
data=Bikedf1, method="ranger")

summarize results

print(model_final2)

```
Random Forest 

10886 samples
    6 predictor

No pre-processing
Resampling: Bootstrapped (25 reps) 
Summary of sample sizes: 10886, 10886, 10886, 10886, 10886, 10886, ... 
Resampling results across tuning parameters:

  mtry  splitrule   RMSE       Rsquared   MAE      
   2    variance    1.2524630  0.7550980  0.9171869
   2    extratrees  1.3181157  0.7456613  0.9787716
  20    variance    0.9663157  0.7682227  0.6771690
  20    extratrees  0.9423590  0.7795570  0.6599200
  38    variance    0.9908111  0.7573062  0.6888095
  38    extratrees  0.9599773  0.7718426  0.6690341

Tuning parameter 'min.node.size' was held constant at a value of 5
RMSE was used to select the optimal model using the smallest value.
The final values used for the model were mtry = 20, splitrule = extratrees an
d min.node.size = 5.
```

summary(model_final2)

```
                          Length Class         Mode     
predictions               10886  -none-        numeric  
num.trees                     1  -none-        numeric  
num.independent.variables     1  -none-        numeric  
mtry                          1  -none-        numeric  
min.node.size                 1  -none-        numeric  
prediction.error              1  -none-        numeric  
forest                        8  ranger.forest list     
splitrule                     1  -none-        character
treetype                      1  -none-        character
r.squared                     1  -none-        numeric  
```

```
call                          9  -none-        call
importance.mode               1  -none-        character
num.samples                   1  -none-        numeric
replace                       1  -none-        logical
xNames                       38  -none-        character
problemType                   1  -none-        character
tuneValue                     3  data.frame    list
obsLevels                     1  -none-        logical
param                         0  -none-        list
```

```
Prediction = exp(predict(model_final2, Bikedf_test))
Bikedf_test$casual = Prediction
head(Bikedf_test)
```

```
             datetime       date month wday hour season holiday workingday weather  temp  atemp humidity windspeed registered    casual
1 2011-01-20 00:00:00 2011-01-20     1    5    0      1       0          1       1 10.66 11.365       56   26.0027  56.720328 3.0793487
2 2011-01-20 01:00:00 2011-01-20     1    5    1      1       0          1       1 10.66 13.635       56    0.0000   7.090939 0.5155312
3 2011-01-20 02:00:00 2011-01-20     1    5    2      1       0          1       1 10.66 13.635       56    0.0000   3.531820 0.4168621
4 2011-01-20 03:00:00 2011-01-20     1    5    3      1       0          1       1 10.66 12.880       56   11.0014   2.527886 0.2321432
5 2011-01-20 04:00:00 2011-01-20     1    5    4      1       0          1       1 10.66 12.880       56   11.0014   2.326857 0.1350988
6 2011-01-20 05:00:00 2011-01-20     1    5    5      1       0          1       1  9.84 11.365       60   15.0013  14.501264 0.1282935
```

```
str(Bikedf_test)
```

```
'data.frame':	6493 obs. of  15 variables:
 $ datetime  : Factor w/ 6493 levels "2011-01-20 00:00:00",..: 1 2 3 4 5 6 7 8 9 10 ...
 $ date      : Date, format: "2011-01-20" "2011-01-20" "2011-01-20" "2011-01-20" ...
 $ month     : Factor w/ 12 levels "1","2","3","4",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ wday      : Factor w/ 7 levels "1","2","3","4",..: 5 5 5 5 5 5 5 5 5 5 ...
 $ hour      : Factor w/ 24 levels "0","1","2","3",..: 1 2 3 4 5 6 7 8 9 10 ...
 $ season    : Factor w/ 4 levels "1","2","3","4": 1 1 1 1 1 1 1 1 1 1 ...
 $ holiday   : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
 $ workingday: Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 2 ...
 $ weather   : Factor w/ 4 levels "1","2","3","4": 1 1 1 1 1 1 1 1 1 2 ...
 $ temp      : num  10.7 10.7 10.7 10.7 10.7 ...
 $ atemp     : num  11.4 13.6 13.6 12.9 12.9 ...
 $ humidity  : int  56 56 56 56 56 60 60 55 55 52 ...
 $ windspeed : num  26 0 0 11 11 ...
 $ registered: num  56.72 7.09 3.53 2.53 2.33 ...
 $ casual    : num  3.079 0.516 0.417 0.232 0.135 ...
```

```
Bikedf_test$count = Bikedf_test$registered + Bikedf_test$casual
head(Bikedf_test)
```

```
             datetime       date month wday hour season holiday workingday weather  temp  atemp humidity windspeed registered    casual
1 2011-01-20 00:00:00 2011-01-20     1    5    0      1       0          1       1 10.66 11.365       56   26.0027  56.720328 3.0793487
2 2011-01-20 01:00:00 2011-01-20     1    5    1      1       0          1       1 10.66 13.635       56    0.0000   7.090939 0.5155312
3 2011-01-20 02:00:00 2011-01-20     1    5    2      1       0          1       1 10.66 13.635       56    0.0000   3.531820 0.4168621
4 2011-01-20 03:00:00 2011-01-20     1    5    3      1       0          1       1 10.66 12.880       56   11.0014   2.527886 0.2321432
5 2011-01-20 04:00:00 2011-01-20     1    5    4      1       0          1       1 10.66 12.880       56   11.0014   2.326857 0.1350988
6 2011-01-20 05:00:00 2011-01-20     1    5    5      1       0          1       1  9.84 11.365       60   15.0013  14.501264 0.1282935
      count
1 59.799677
2  7.606471
3  3.948682
4  2.760029
5  2.461955
6 14.629558
```

```
Bikedf_test1 = Bikedf_test
Bikedf_test$count = Bikedf_test$registered + Bikedf_test$casual
head(Bikedf_test)
Bikedf_test1 = Bikedf_test
```

#Calculate money due to parent company, including 2.5% penalty.

```
Bikedf_test1$registered_owed = Bikedf_test1$registered * 1.50*.25 * 1.025
Bikedf_test1$casual_owed = Bikedf_test1$casual*2.25 *.25 * 1.025
head(Bikedf_test1)
```

```
             datetime season     holiday workingday weather  temp  atemp humidity windspeed       date month wday hour registered    casual
1 2011-01-20 00:00:00 Winter Regular_day Workingday    Good 10.66 11.365       56   26.0027 2011-01-20     1    5    0   76.71265 3.3101578
2 2011-01-20 01:00:00 Winter Regular_day Workingday    Good 10.66 13.635       56    0.0000 2011-01-20     1    5    1    0.00000 0.0000000
3 2011-01-20 02:00:00 Winter Regular_day Workingday    Good 10.66 13.635       56    0.0000 2011-01-20     1    5    2    0.00000 0.7354419
4 2011-01-20 03:00:00 Winter Regular_day Workingday    Good 10.66 12.880       56   11.0014 2011-01-20     1    5    3    0.00000 0.0000000
5 2011-01-20 04:00:00 Winter Regular_day Workingday    Good 10.66 12.880       56   11.0014 2011-01-20     1    5    4    0.00000 0.0000000
6 2011-01-20 05:00:00 Winter Regular_day Workingday    Good  9.84 11.365       60   15.0013 2011-01-20     1    5    5    0.00000 0.0000000
       count registered_owed casual_owed
1 80.0228043        29.48642   1.9085128
2  0.0000000         0.00000   0.0000000
3  0.7354419         0.00000   0.4240282
4  0.0000000         0.00000   0.0000000
5  0.0000000         0.00000   0.0000000
6  0.0000000         0.00000   0.0000000
```

Results:

```
registered_owed = round(sum(Bikedf_test1$registered_owed),2)
casual_owed = round(sum(Bikedf_test1$casual_owed),2)
total_owed = registered_owed+casual_owed
paste0("The amounts owing, including the penalty, that need to be paid for registered and
casual riders respectively are $",
      registered_owed, " and $", casual_owed,", for a total of $", total_owed, ".")
```

```
[1] "The amounts owing, including the penalty, that need to be paid for
registered and casual riders respectively are $351,874.64 and $110,881.85,
for a total of $462,756.49."
```

In order to test the results, we compared the total owing in the Bikedf1_test dataset to the total paid in the original Bikedf1 dataset with the following code:

```
Bikedf1$registered_owed = Bikedf1$registered * 1.50*.25 * 1.025
Bikedf1$casual_owed = Bikedf1$casual*2.25 *.25 * 1.025
registered_owed.Bikedf1 = round(sum(Bikedf1$registered_owed),2)
casual_owed.Bikedf1 = round(sum(Bikedf1$casual_owed),2)
total_owed.Bikedf1 = registered_owed.Bikedf1+casual_owed.Bikedf1
paste0("The amounts that were paid on the original train file, including the penalty (just for comp
arison purposes),  for registered and casual riders respectively are $",
      registered_owed.Bikedf1, " and $", casual_owed.Bikedf1,", for a total of $", total_owed.Bike
df1, ".")
```

```
[1] "The amounts that were paid on the original train file, including the
penalty (just for comparison purposes), for registered and casual riders
respectively are $650,877.95 and $226,090.34, for a total of $876,968.29."
```

The average revenue per datapoint in Bikedf1_test is $71.27 ($462,756.49/6,493 entries) while the comparable average for the Bikedf1 dataset is $80.56 ($876,968.29/10,886 entries). This tells us that our predictions for the revenue owed to the parent are a reasonable estimate with what had been paid where the full data was available.

Deployment

Several models were evaluated, however, based on the above analysis, the XGBLinear model will predict registered ridership with the greatest accuracy while the RandomForest model with predict the casual ridership with the greatest accuracy. Initial test results indicate this model is 98.9% accurate based on projected payments to historical payments.

Future Enhancements / Next Steps
Comparing data from other cities with a bike sharing program would be the most useful. Washington's' climate is in a "goldilocks" zone where it's not too hot or cold and the number of days of extreme weather is minimal. Climates with more snow as seen in Canadian cities or in the south Florida may not embrace bike sharing as seen by Washington. Also, cities that are based on a "car culture" may be reluctant to use a bike sharing program.

Data on some of the demographics of the uses may be useful in promoting the service in Washington and other cities as the service expands.

Weather data should be collected from the national weather service for improved accuracy and to save time recording the hourly weather.

Demographics on registered riders would assist in marketing to new riders.

Rental location data would be helpful to determine the most popular locations.

For registered riders having their residence addresses would also be helpful to see the geographic region that riders are attracted from.