CSC-423 PROJECT SUBMISSION

Application of Linear Regression on
Kaggle Bike Sharing Dataset
Capital Bike share program,
Washington D.C., USA

(Technical Report)

by

Pradeep Sathyamurthy

Under the guidance of: Prof. Nandhini Gulasingam
DePaul University

23rd November 2016

Contents

Project Summary:

Introduction:

Bike sharing systems are new generation of traditional bike rentals where whole process from membership, rental and return back has become automatic. Through these systems, user is able to easily rent a bike from a particular position and return back at another position. Currently, there are over ***500 bike-sharing programs around the world*** which is composed of over 500 thousand bicycles. Today, there exists great interest in these systems due to their important role in traffic, environmental and health issues.

Apart from interesting real world applications of bike sharing systems, the ***characteristics of data being generated by these systems make them attractive for the research***. Opposed to other transport services such as bus or subway, the duration of travel, departure and arrival position is explicitly recorded in these systems. ***This feature turns bike sharing system into a virtual sensor network that can be used for sensing mobility in the city***.

Project Scope:

In this project our ***main interest*** is to ***find the count of bikes required for a particular time in a day to keep up the user demand.*** For this research ***we will scope our algorithms within Linear Regression methodology***. Kaggle [1] shows that applying more complex algorithms like SVM, Neural Networks, gradient boost, etc., can increase the efficiency of the final model. However, we will limit our research for the algorithms thought as part of course CSC-423 which we have used.

Dataset Description:

The core data set is related to the two-year historical log corresponding to years 2011 and 2012 from ***Capital Bike share system***, Washington D.C., USA which is publicly available [2]. Dataset is aggregated hourly and daily basis and then extracted and added with the corresponding weather [3] and seasonal information.

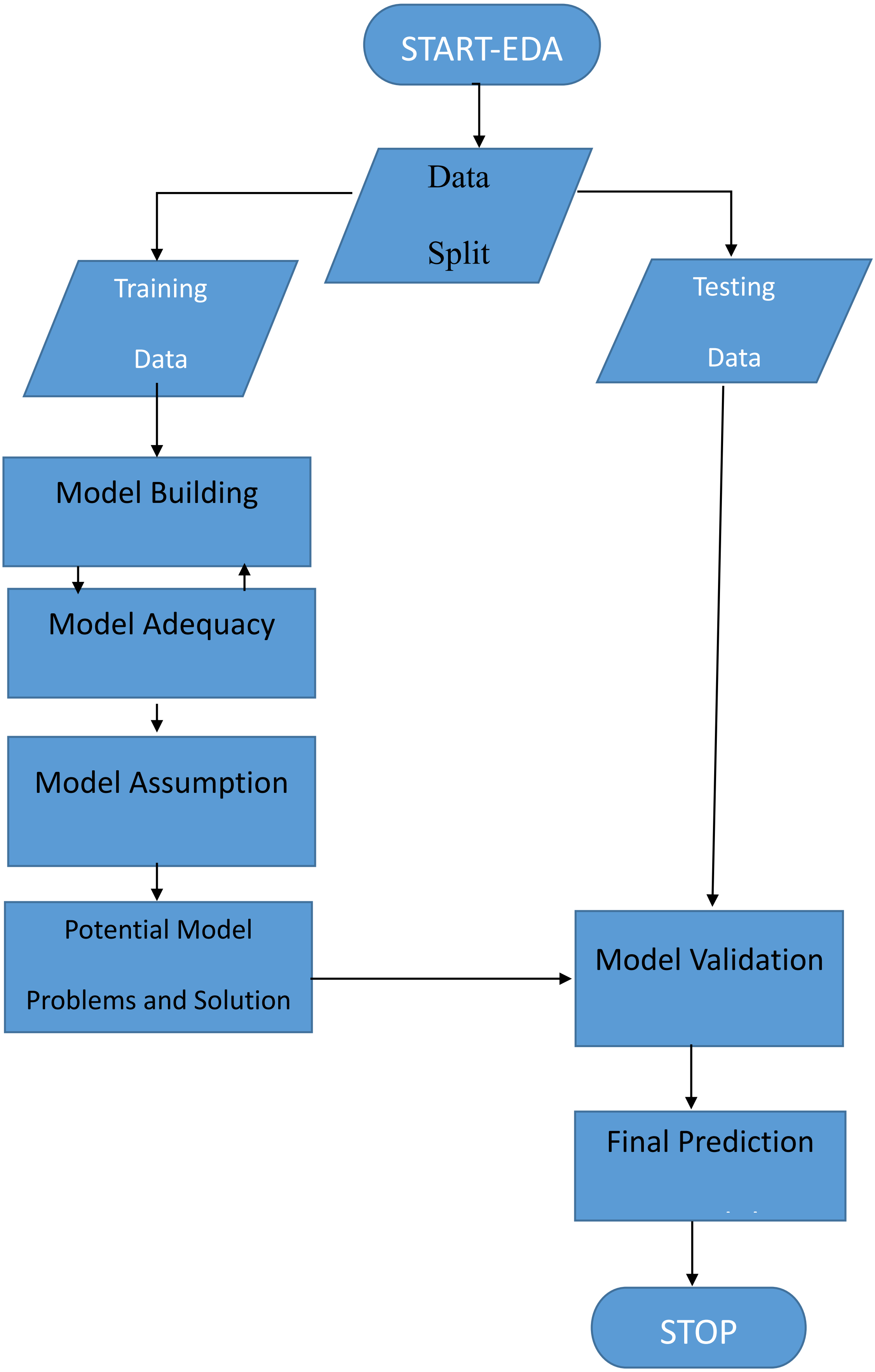
Files:

We were shared with two files train.csv and test.csv. Former to train the algorithm and later to validate the final algorithm built which do not have the values for dependent variable (count) for competition evaluation purpose by Kaggle. However, for this project since we are requested to show and prove the final model behaviour, we have ***considered only train.csv file*** which has the information of dependent variable. Based on this train.csv file only ***we created our Train [8710 records] and Test data [2176 records]*** for model building and model validation respectively. Thus, for this project purpose test.csv file shared by Kaggle has been discarded. Below are features descriptions available as part of the dataset:

SI.NO	Variable Name	Description
1	Datetime[T]	hourly date + timestamp
2	Season[C]	1 = spring, 2 = summer, 3 = fall, 4 = winter
3	holiday[C]	whether the day is considered a holiday
4	workingday[C]	whether the day is neither a weekend nor holiday
5	weather[C]	1: Clear, Few clouds, Partly cloudy, Partly cloudy 2: Mist + Cloudy, Mist + Broken clouds, Mist + Few clouds, Mist 3: Light Snow, Light Rain + Thunderstorm + Scattered clouds, Light Rain + Scattered clouds 4: Heavy Rain + Ice Pallets + Thunderstorm + Mist, Snow + Fog
6	temp[N]	temperature in Celsius
7	atemp[N]	"feels like" temperature in Celsius
8	humidity[N]	relative humidity
9	windspeed[N]	wind speed
10	*casual[N]*	*number of non-registered user rentals initiated*
11	*registered[N]*	*number of registered user rentals initiated*
12	***count[N]***	***number of total rentals i.e. [casual + registered]***

[T]=Time Series, [C] = Categorical, [N] = Numerical/Quantitative

Project Methodology Layout



Exploratory Data Analysis

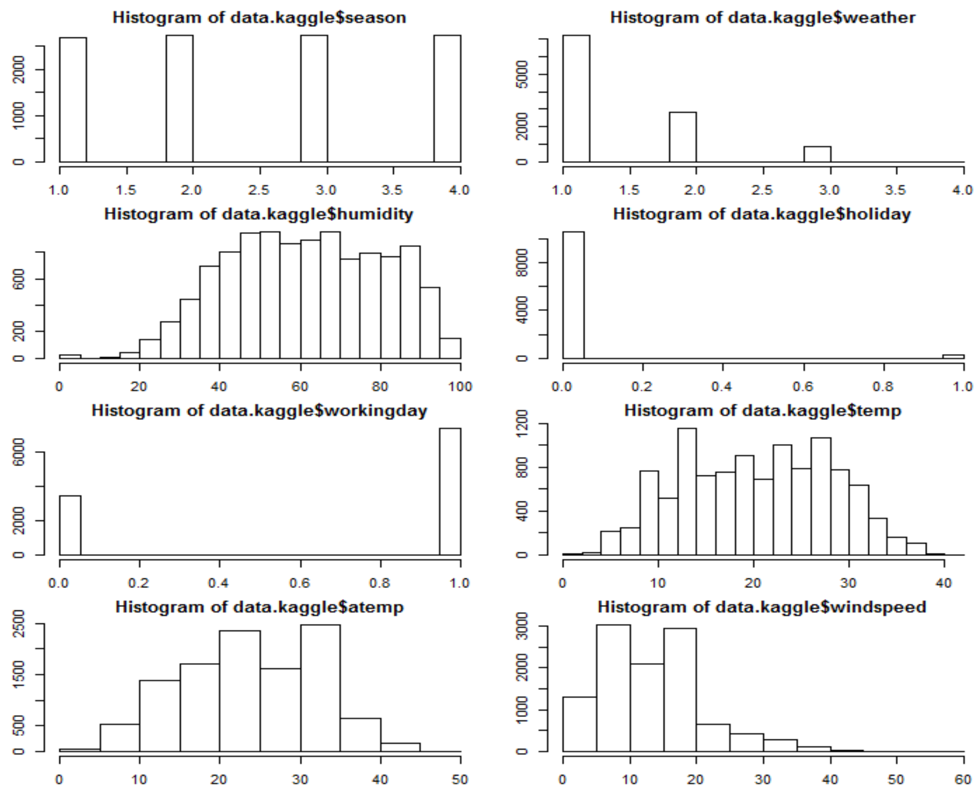
It is always wise to start a model building with hypothesis generation and an exploratory data analysis. This will help us to:

- Understand the relationship between the variables
- Gain domain expertise
- Avoid bias based samples
- Build a structure modelling with a structured approach

Thus we hypothesized few scenarios based on our dataset, I would like to highlight couple of them here:

1. High usage time would be during work commuting hours (7-9AM and 4-6PM)
2. Weather should have significant impact on bike usage (Rain/snow=less usage, normal weather=more usage)

Best approach to validate these hypothesis is through visualization, below are few EDA[R] done on ***train.csv*** file:



1. ***Season*** have 4 levels with equal distribution, this infers their impact in building model would be ***less***

2. ***Weather***-1 = Clear Weather has higher ride counts, this infers their impact in building model will be ***significant***

3. ***Humidity and Temperature*** Normally Distributed, this infers their impact in building model will be ***significant***

4. High usage by ***Subscribers*** during ***weekdays*** and high usage by ***Customers*** during ***weekends*** or holidays

5. ***Tem and Atemp*** have similar distribution, there is a chance ***multicollinearity***, can be confirmed based on VIF values

6. ***Wind speed*** is skewed towards right, this infers their impact in building model will be ***less***, can be confirmed with correlation values further

Data Split

In this dataset we totally have 4 Categorical variables with 2 variables (holiday and working day) being a binary data, while other 2 variables (season and weather) are each with 4 levels. Which demands totally 6 dummy variables. Thus, superficially we can assume that there are totally 8 categorical variables (2 binary + 6 dummy). With respective to quantitative variables we have 4 variables (temp, atemp, humidity and winds peed). So, totally we have 12 variables (8 categorical + 4 quantitative). As per thumb of rule we need to have at least 120 sample records in order to split the data in to train and test data. Since, we have totally 10886 records as part of our train.csv we can do a split[R] of ***Training-Data: Testing-Data = 80:20 ratios*** through which ***Training data can be used to build our model while Testing Data can be used to test our model for model validation and prediction***.

Training Data:

From here on we will refer our Training dataset with name ***data.train*** which is 80% of Simple Random Sampled data from train.csv file. Data.train has a total number of 8710 samples through which we will be building and training our model.

Training Dataset: data.train = 80% of (10886 samples of train.csv file)

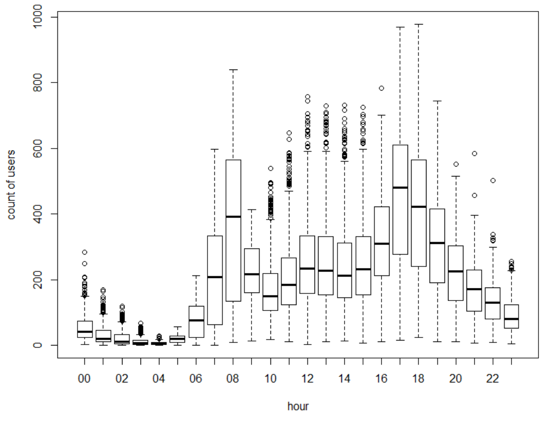
Test Data:

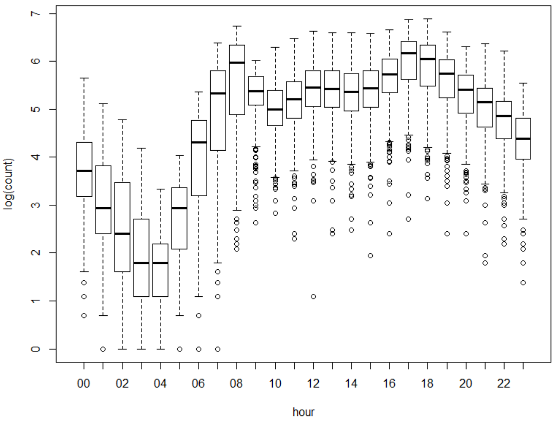
From here on we will refer our Testing dataset with name ***data.test*** which is the remaining 20% of Simple Random Sampled data from train.csv file. Data.test has a total number of 2176 samples through which we will be validating our model and subject it for prediction. Since, our testing dataset is a remaining sample left out by training data, ***our data.test is no way a subset of data.train these are two simple random samples of train.csv***.

Testing Dataset: data.train = 20% of (remaining 10886 samples of train.csv file)

Other Data Clean-up activities:

Validation for missing values have been done and ***we see no missing values in data***.





- Based on the histograms and boxplot laid for each variable we observed few outliers in the dataset.
- However, these data are generated by sensors and hence there is no chance for manual error occurring. So for model building activity, we would like to ***consider them as natural outliers***. Possible reasons could be group of users (friends) hiring cycle at point of time.
- We assumed Log2 can fix this better and it worked. In the above figure image on left is without transformation and image on right is with log transformation on dependent variable "count". This also ***hint log transform on final model*** can give great improvement to our model.
- Also, from our dataset we understood that ***dependent variable count = Number of Registered + Casual users***. So we can exclude variables "registered" and "casual" from our data.train and data.test for model building or validation.

Model Building:

Excluding time series data – Datetime variable

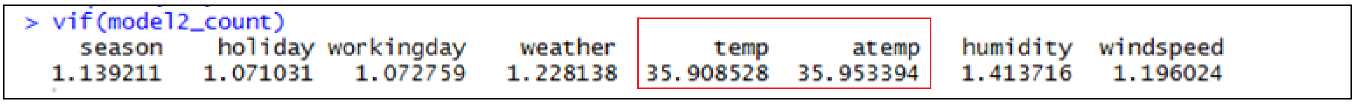
We first started to build our model[R] with variables which are categorical and Quantitative in nature. Since variable datatime of data.train was continuous time series data, we excluded the same for our initial model building activity.

Variable Reduction

Before we proceed with model building, we wanted to check if there are any correlated variables within independent variables. Based on that we wanted to build a model including and excluding them and wanted to derive an efficient model.

	train.registered	train.casual	train.count	train.temp	train.humidity	train.atemp	train.windspeed
train.registered	1	0.49724969	0.9709481	0.31857128	-0.26545787	0.31463539	0.09105166
train.casual		1	0.6904136	0.46709706	-0.3481869	0.46206654	0.09227619
train.count			1	0.39445364	-0.31737148	0.38978444	0.10136947
train.temp				1	-0.06494877	0.98494811	-0.01785201
train.humidity					1	-0.04353571	-0.31860699
train.atemp						1	-0.057473
train.windspeed							1

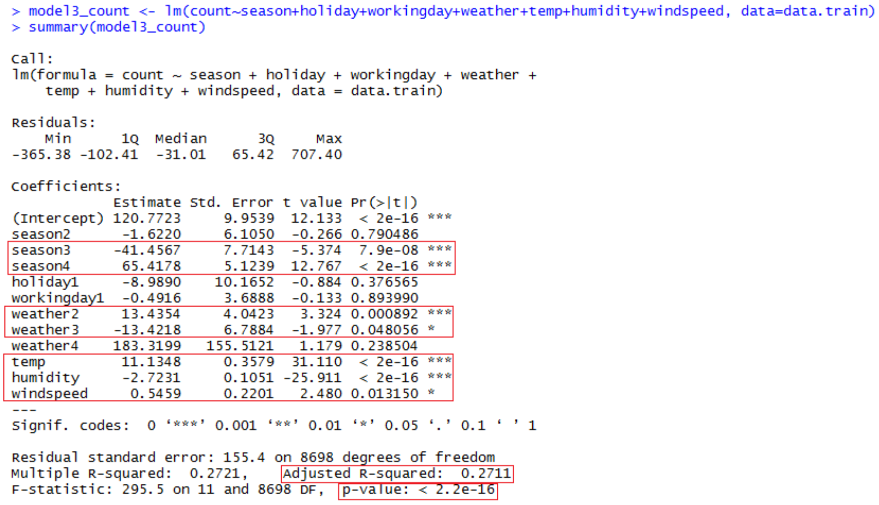
From above table we observed that independent variables ***temp and atemp had a high correlation of 0.985***. We studied domain significance which said Temp is an actual temperature while atemp is a feel-like temperature in atmosphere.



```
> vif(model2_count)
   season   holiday workingday    weather       temp      atemp   humidity  windspeed
 1.139211  1.071031   1.072759   1.228138  35.908528  35.953394   1.413716   1.196024
```

Based on this study and Variance inflation factors shown above we finalized to ***remove one variable which was atem*** for our rest of the model building activity. Rest of the variables where subjected for model building to see the variance explained by them:

Basic Model Building



```
> model3_count <- lm(count~season+holiday+workingday+weather+temp+humidity+windspeed, data=data.train)
> summary(model3_count)

Call:
lm(formula = count ~ season + holiday + workingday + weather + 
    temp + humidity + windspeed, data = data.train)

Residuals:
    Min      1Q  Median      3Q     Max 
-365.38 -102.41  -31.01   65.42  707.40 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 120.7723     9.9539  12.133  < 2e-16 ***
season2      -1.6220     6.1050  -0.266 0.790486    
season3     -41.4567     7.7143  -5.374  7.9e-08 ***
season4      65.4178     5.1239  12.767  < 2e-16 ***
holiday1     -8.9890    10.1652  -0.884 0.376565    
workingday1  -0.4916     3.6888  -0.133 0.893990    
weather2     13.4354     4.0423   3.324 0.000892 ***
weather3    -13.4218     6.7884  -1.977 0.048056 *  
weather4    183.3199   155.5121   1.179 0.238504    
temp         11.1348     0.3579  31.110  < 2e-16 ***
humidity     -2.7231     0.1051 -25.911  < 2e-16 ***
windspeed     0.5459     0.2201   2.480 0.013150 *  
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 155.4 on 8698 degrees of freedom
Multiple R-squared:  0.2721,	Adjusted R-squared:  0.2711 
F-statistic: 295.5 on 11 and 8698 DF,  p-value: < 2.2e-16
```

- Over all model was significant with p-value < 0.01
- Variables season, weather, temp, humidity and wind-speed seem significant variables with respective p-value < 0.05
- However, ***the variance that this model could explain was only 27.11% which is considered to be not efficient***
- This ***demands building few feature engineering*** and model efficiency improvement in analysis

Including Time series data – Datetime variable

Since the basic model with all variables included except Datetime variable explained only 25.28% of variance in identifying count of bicycles needed, we wanted to build a model where there is a participation from Datetime variable too. However, this variable cannot be used directly due to its nature e.g. `"2011-01-01 00:00:00".` Thus we decided to do some feature engineering on this variable.

Feature Engineering – Binning

Feature engineering is one of the main step towards building an efficient model. Binning[R] is one such feature engineering which ***introduce more variables to improve the model***. Thus, for our dataset we need to Bin datetime variable into separate sub-strings. That is, we want to split Datetime variable into:

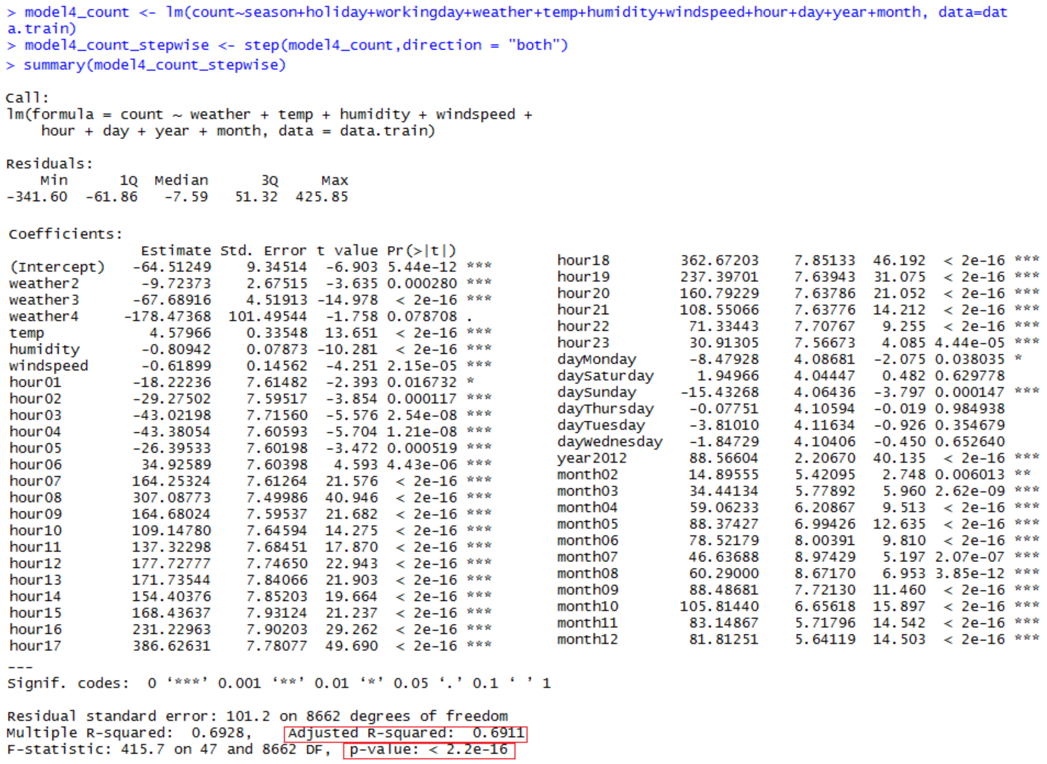
1. Hours
2. Weekdays
3. Months
4. Year

By doing this we divided our 1 time-series variable Datetime into 4 categorical variables:

Sample Datetime data	Hours	Weekdays	Month	Year
2011-01-01 00:00:00	01	Saturday	January	2011

Modelling after Feature Engineering

We subjected all variables along with new features obtained above from Datetime variable to build a model and then use this model in variable screening algorithm[R] like stepwise and backwards selection to choose the most significant variables from data:



```
> model4_count <- lm(count~season+holiday+workingday+weather+temp+humidity+windspeed+hour+day+year+month, data=dat
a.train)
> model4_count_stepwise <- step(model4_count,direction = "both")
> summary(model4_count_stepwise)

Call:
lm(formula = count ~ weather + temp + humidity + windspeed + 
    hour + day + year + month, data = data.train)

Residuals:
    Min      1Q  Median      3Q     Max 
-341.60  -61.86   -7.59   51.32  425.85 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept)  -64.51249    9.34514  -6.903 5.44e-12 ***
weather2      -9.72373    2.67515  -3.635 0.000280 ***
weather3     -67.68916    4.51913 -14.978  < 2e-16 ***
weather4    -178.47368  101.49544  -1.758 0.078708 .  
temp           4.57966    0.33548  13.651  < 2e-16 ***
humidity      -0.80942    0.07873 -10.281  < 2e-16 ***
windspeed     -0.61899    0.14562  -4.251 2.15e-05 ***
hour01       -18.22236    7.61482  -2.393 0.016732 *  
hour02       -29.27502    7.59517  -3.854 0.000117 ***
hour03       -43.02198    7.71560  -5.576 2.54e-08 ***
hour04       -43.38054    7.60593  -5.704 1.21e-08 ***
hour05       -26.39533    7.60198  -3.472 0.000519 ***
hour06        34.92589    7.60398   4.593 4.43e-06 ***
hour07       164.25324    7.61264  21.576  < 2e-16 ***
hour08       307.08773    7.49986  40.946  < 2e-16 ***
hour09       164.68024    7.59537  21.682  < 2e-16 ***
hour10       109.14780    7.64594  14.275  < 2e-16 ***
hour11       137.32298    7.68451  17.870  < 2e-16 ***
hour12       177.72777    7.74650  22.943  < 2e-16 ***
hour13       171.73544    7.84066  21.903  < 2e-16 ***
hour14       154.40376    7.85203  19.664  < 2e-16 ***
hour15       168.43637    7.93124  21.237  < 2e-16 ***
hour16       231.22963    7.90203  29.262  < 2e-16 ***
hour17       386.62631    7.78077  49.690  < 2e-16 ***
hour18         362.67203    7.85133  46.192  < 2e-16 ***
hour19         237.39701    7.63943  31.075  < 2e-16 ***
hour20         160.79229    7.63786  21.052  < 2e-16 ***
hour21         108.55066    7.63776  14.212  < 2e-16 ***
hour22          71.33443    7.70767   9.255  < 2e-16 ***
hour23          30.91305    7.56673   4.085 4.44e-05 ***
dayMonday       -8.47928    4.08681  -2.075 0.038035 *  
daySaturday      1.94966    4.04447   0.482 0.629778    
daySunday      -15.43268    4.06436  -3.797 0.000147 ***
dayThursday     -0.07751    4.10594  -0.019 0.984938    
dayTuesday      -3.81010    4.11634  -0.926 0.354679    
dayWednesday    -1.84729    4.10406  -0.450 0.652640    
year2012        88.56604    2.20670  40.135  < 2e-16 ***
month02         14.89555    5.42095   2.748 0.006013 ** 
month03         34.44134    5.77892   5.960 2.62e-09 ***
month04         59.06233    6.20867   9.513  < 2e-16 ***
month05         88.37427    6.99426  12.635  < 2e-16 ***
month06         78.52179    8.00391   9.810  < 2e-16 ***
month07         46.63688    8.97429   5.197 2.07e-07 ***
month08         60.29000    8.67170   6.953 3.85e-12 ***
month09         88.48681    7.72130  11.460  < 2e-16 ***
month10        105.81440    6.65618  15.897  < 2e-16 ***
month11         83.14867    5.71796  14.542  < 2e-16 ***
month12         81.81251    5.64119  14.503  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 101.2 on 8662 degrees of freedom
Multiple R-squared:  0.6928,	Adjusted R-squared:  0.6911 
F-statistic: 415.7 on 47 and 8662 DF,  p-value: < 2.2e-16
```

- Stepwise, Forward and backward selection algorithms gave same output as above
- Over all model was significant with p-value < 0.01
- Variables weather, temp, humidity, windspeed, hour, day, year and month seems to be significant
- Adjusted R-square variable has been ***highly improved from 0.2711 in our previous model[M] to 0.6911*** in this model
- Thus, model after Binning improves the efficiency of our model which now ***explains 69.11% of variance*** in count of bicycles needed.

Model Adequacy

Now that we have built a model which explains 69.11% variance in count of bicycles needed, we need to see the adequacy of these models in order to predict the count of bicycles. As part of this model adequacy, based on the feature selection done as part of above model selection[M] we will try with few interaction and polynomial terms and study the adequacy of the model based on:

1. Global F- Test
2. Beta Significance T-Test
3. Residual Error
4. Root Mean Square Error

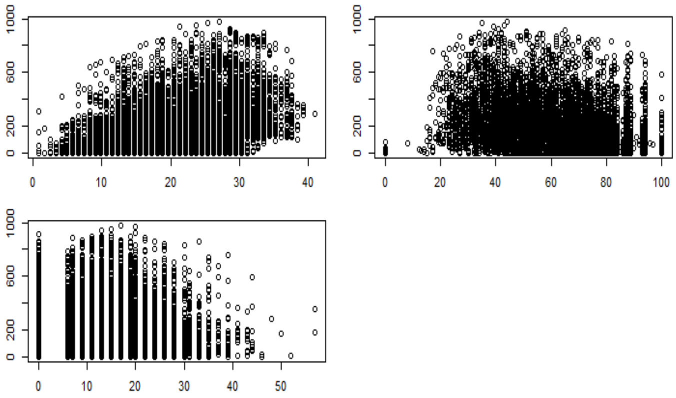
Identifying pattern through visualization

Graph on the right side shows a scatter plot of count variable (independent variable) with few of the significant variables chosen as part of feature selection.

First row shows relation between count with temperature and humidity, while second row shows the relation between count and windspeed.

From these graph we couldn't see any U-shape and hence there is a ***less possibility*** of model improvement due to ***polynomial terms***.

Also, there are no correlation that exist within temp, humidity and windspeed as seen above[C],hence there is a ***less possibility*** of model improvement due to ***interaction terms*** as well.

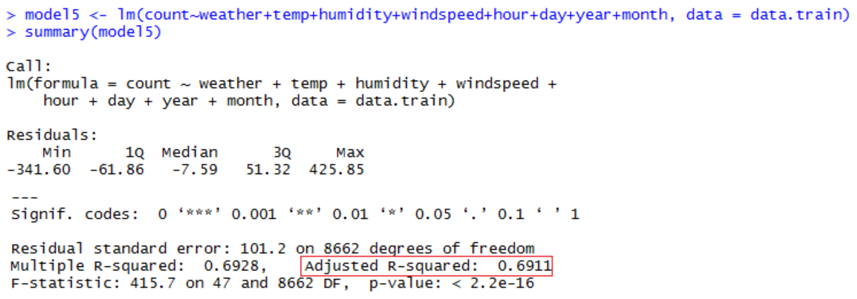


First Order Model

This is the first order simple regression model with feature engineering done.

The overall model is significant in explaining the variance of count of bicycles needed.

Adj-R^2 shows it explain ***69.11%*** of variance in explaining the count of bicycles needed



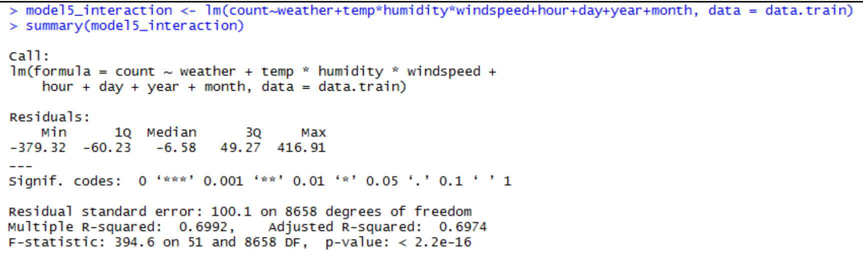
```
> model5 <- lm(count~weather+temp+humidity+windspeed+hour+day+year+month, data = data.train)
> summary(model5)

Call:
lm(formula = count ~ weather + temp + humidity + windspeed +
    hour + day + year + month, data = data.train)

Residuals:
    Min      1Q  Median      3Q     Max
-341.60  -61.86   -7.59   51.32  425.85

 ---
 Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

 Residual standard error: 101.2 on 8662 degrees of freedom
 Multiple R-squared:  0.6928,    Adjusted R-squared:  0.6911
 F-statistic: 415.7 on 47 and 8662 DF,  p-value: < 2.2e-16
```

Interaction Model

This is a complete interaction model built with significant quantitative variable.

Overall model is significant.

However, there is only a bleak improvement in Adj-R^2 value by explaining ***69.74%*** of variance in Count.



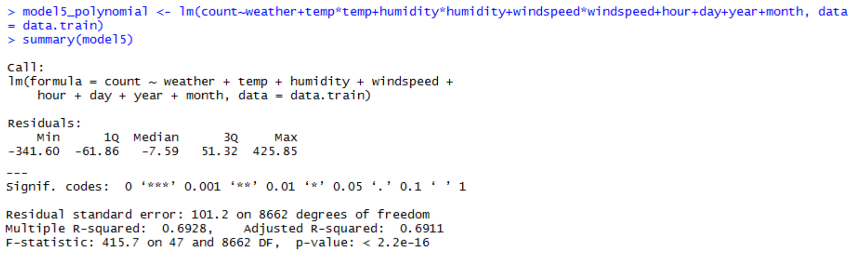
```
> model5_interaction <- lm(count~weather+temp*humidity*windspeed+hour+day+year+month, data = data.train)
> summary(model5_interaction)

Call:
lm(formula = count ~ weather + temp * humidity * windspeed +
    hour + day + year + month, data = data.train)

Residuals:
    Min      1Q  Median      3Q     Max
-379.32  -60.23   -6.58   49.27  416.91
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 100.1 on 8658 degrees of freedom
Multiple R-squared:  0.6992,    Adjusted R-squared:  0.6974
F-statistic: 394.6 on 51 and 8658 DF,  p-value: < 2.2e-16
```

Polynomial Model

This is a complete polynomial model built with significant quantitative variables.

Overall model is significant.

However, it doesn’t increase the Adj-R^2 of the model, even this explains ***69.11%*** of variance in count as simple liner model does.



```
> model5_polynomial <- lm(count~weather+temp*temp+humidity*humidity+windspeed*windspeed+hour+day+year+month, data
= data.train)
> summary(model5)

Call:
lm(formula = count ~ weather + temp + humidity + windspeed + 
    hour + day + year + month, data = data.train)

Residuals:
    Min      1Q  Median      3Q     Max 
-341.60  -61.86   -7.59   51.32  425.85 

---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 101.2 on 8662 degrees of freedom
Multiple R-squared:  0.6928,	Adjusted R-squared:  0.6911 
F-statistic: 415.7 on 47 and 8662 DF,  p-value: < 2.2e-16
```

Model Adequacy Details for all models built:

SI.NO	MODEL	Number of Betas	F-Test	T-Test (Number of Significant Beta)	ADJ R-SQ	RMSE
			First Order Model			
1	Model3[M]	11	Significant	7	0.2711	155.4
2	***Model5[M]***	***47***	***Significant***	***42***	***0.6911***	***101.2***
			Interaction Model			
2	Model5_interaction[M]	72	Significant	42	0.6982	100
			Quadratic Model			
3	Model5_polynomial[M]	48	Significant	42	0.6911	101.2

From above table, it is very clear that ***Interaction or polynomial models haven’t helped in increasing the model efficiency***, thus first order linear multiple regression model with feature engineering (model5) seems more optimal for predicting count of bicycles for below reasons:

- **Highest Adjusted R-Square with less predictors = 69.11%**
- **Less Number of Predictors (Betas)**
- **Overall model is Significant in F-Test**
- **Beta seems significant in T-Test**
- **Lowest RMSE with less predictors = 101.2**

Now that we have concluded that simple linear model with feature engineering done is the optimal model, we will subject it to check for model assumptions and confirm if it satisfied our assumption

Model Subjected to residual checks

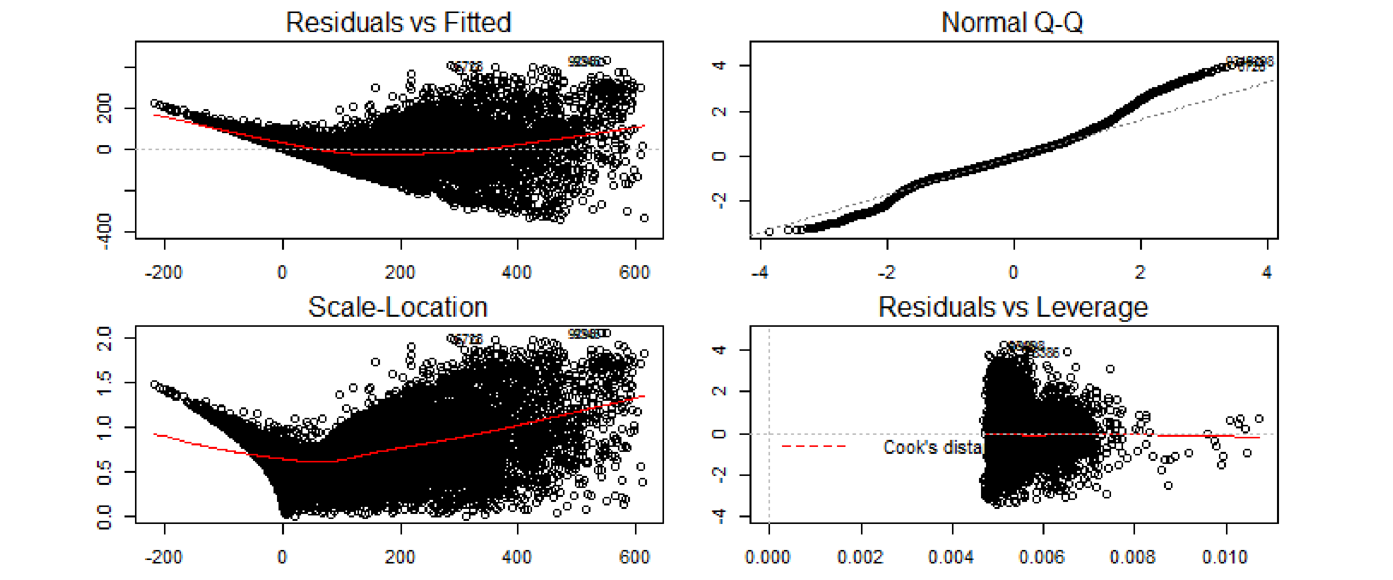
*count~**weather+temp+humidity+windspeed+hour+day+year+month***

Model Assumption

We start to create any models with few assumptions, in which two major assumptions are:

1. **All pairs of error terms are not correlated that is error terms are independent to each other**
2. **Error is normally distributed with mean=0 and Standard Deviation being constant**

So in order to re-confirm that our assumptions hold good for the model that would be subjected to prediction, we need to perform few residual analyses before concluding the final model.



Residual Analysis

1. Graph plotted between Residual and Fitted is used to confirm assumption-1
2. We expect residual plot with no trends or pattern. From above figure we could infer that ***there is a concrete trend that exists*** in this plot.
3. ***There is a dramatic increase in variability***, as a thumb of rule not more than 5% of the residual should be more than 2*(Standard Deviation) of error above or below zero. **5% of (8710) is 435,** thus we should not see more than 435 points that lie outside **(2 * (182.0533)=364)** 364. From above figure we could infer that there are more points that lie above 2-standard deviation. Thus, we can conclude that our assumption concerning to error being independent is not satisfied.

Heteroscedasticity:

1. We say a model as heteroscedastic when there is no constant variance. Funnel shape of residual plot clearly identifies the model is heteroscedastic.
2. Even in our residual graph we see a funnel shape and can say that ***our model is heteroscedastic*** in nature.

Normal Probability Plot:

1. From normality plot for the residual, we can notice that most of the points fall reasonably close to straight line which indicates that ***normality assumption is satisfied***. However, there is an indication of s-shape

Outlier and influential Points:

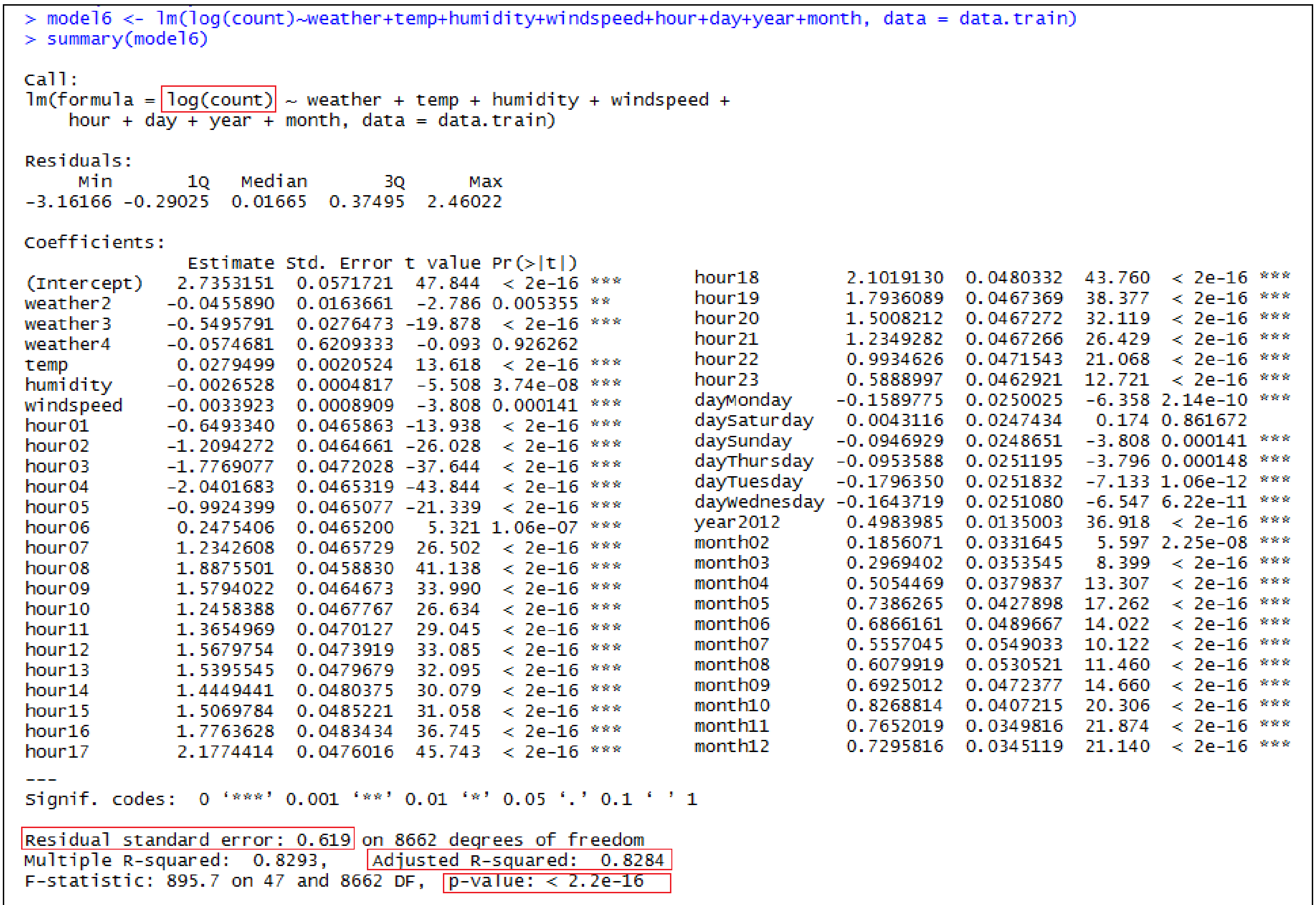
1. Residual vs Leverage graph infers that there are few influential or outliers present.
2. Hence, we calculated for observations which are considered as outliers based on model built, any studentized residual greater than 3 or less than -3 where considered as outliers. We obtained a list of 65 observations.
3. We also wanted to find observations which are influential based on H-hat method. We got a cut off of 0.01125144 and hence considered any value above this as influential point. However, our model fetched only one observation.
4. We compared list of outliers with influential point and there was no match and hence concluded that these are ***natural outliers and removing them will either over-fit or under fit the model***.

Potential Model Problem and Solution:

Above residual analysis clearly indicates that our model is suffering from heteroscedasticity, that is a state with non-constant variance. Thus, we cannot use this model directly for prediction or model validation. We need to fix this. One possible solution is to try transforming the dependent variable and see how our model behaves with respective to explaining variance and residual behaviour. So, we tried to perform model transformation on model selected from model adequacy for residual analysis[M].

Model Transformation:

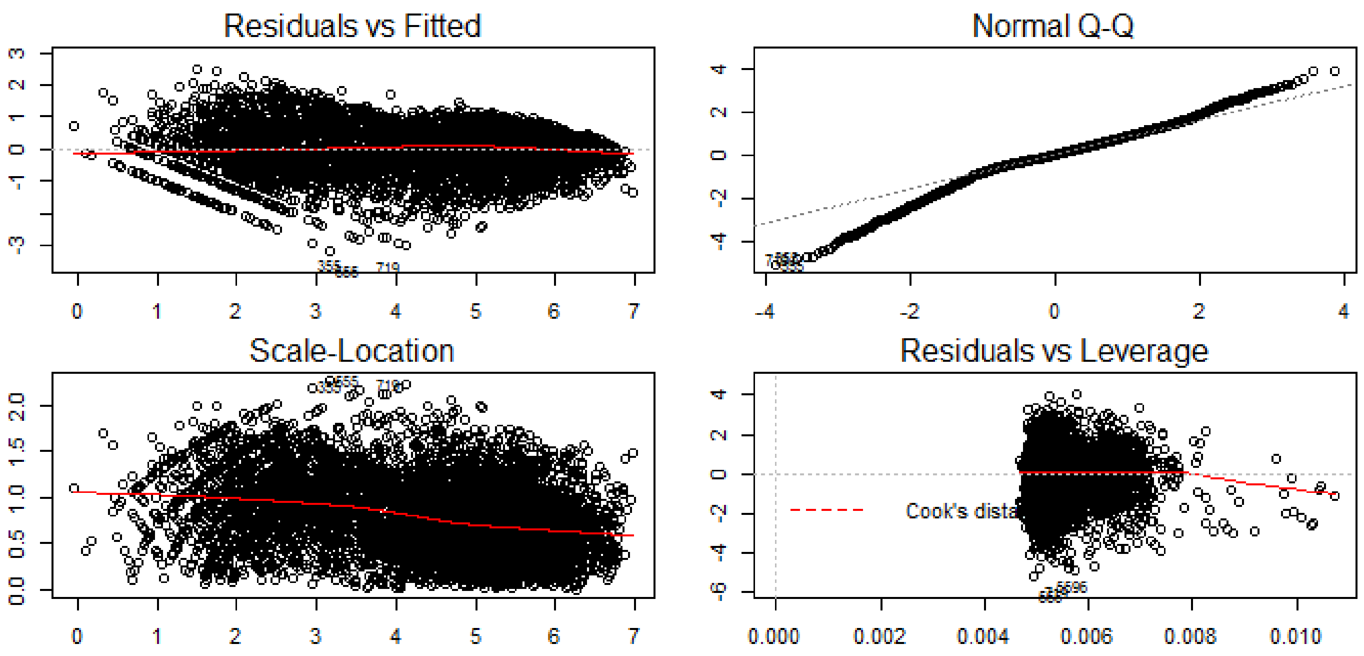
Since the normal probability plot show a s-shape, we assumed log transformation will do a great difference. Further based on EDA[E] done, we found that **doing log transformation on count had a significant impact** on outliers/influential points and changed the whole shape of box-plot. This confirmed us to try with log transformation for our model[M].



```
> model6 <- lm(log(count)~weather+temp+humidity+windspeed+hour+day+year+month, data = data.train)
> summary(model6)

Call:
lm(formula = log(count) ~ weather + temp + humidity + windspeed + 
    hour + day + year + month, data = data.train)

Residuals:
     Min       1Q   Median       3Q      Max 
-3.16166 -0.29025  0.01665  0.37495  2.46022 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept)  2.7353151  0.0571721  47.844  < 2e-16 ***
weather2    -0.0455890  0.0163661  -2.786 0.005355 ** 
weather3    -0.5495791  0.0276473 -19.878  < 2e-16 ***
weather4    -0.0574681  0.6209333  -0.093 0.926262    
temp         0.0279499  0.0020524  13.618  < 2e-16 ***
humidity    -0.0026528  0.0004817  -5.508 3.74e-08 ***
windspeed   -0.0033923  0.0008909  -3.808 0.000141 ***
hour01      -0.6493340  0.0465863 -13.938  < 2e-16 ***
hour02      -1.2094272  0.0464661 -26.028  < 2e-16 ***
hour03      -1.7769077  0.0472028 -37.644  < 2e-16 ***
hour04      -2.0401683  0.0465319 -43.844  < 2e-16 ***
hour05      -0.9924399  0.0465077 -21.339  < 2e-16 ***
hour06       0.2475406  0.0465200   5.321 1.06e-07 ***
hour07       1.2342608  0.0465729  26.502  < 2e-16 ***
hour08       1.8875501  0.0458830  41.138  < 2e-16 ***
hour09       1.5794022  0.0464673  33.990  < 2e-16 ***
hour10       1.2458388  0.0467767  26.634  < 2e-16 ***
hour11       1.3654969  0.0470127  29.045  < 2e-16 ***
hour12       1.5679754  0.0473919  33.085  < 2e-16 ***
hour13       1.5395545  0.0479679  32.095  < 2e-16 ***
hour14       1.4449441  0.0480375  30.079  < 2e-16 ***
hour15       1.5069784  0.0485221  31.058  < 2e-16 ***
hour16       1.7763628  0.0483434  36.745  < 2e-16 ***
hour17       2.1774414  0.0476016  45.743  < 2e-16 ***
hour18        2.1019130  0.0480332  43.760  < 2e-16 ***
hour19        1.7936089  0.0467369  38.377  < 2e-16 ***
hour20        1.5008212  0.0467272  32.119  < 2e-16 ***
hour21        1.2349282  0.0467266  26.429  < 2e-16 ***
hour22        0.9934626  0.0471543  21.068  < 2e-16 ***
hour23        0.5888997  0.0462921  12.721  < 2e-16 ***
dayMonday    -0.1589775  0.0250025  -6.358 2.14e-10 ***
daySaturday   0.0043116  0.0247434   0.174 0.861672    
daySunday    -0.0946929  0.0248651  -3.808 0.000141 ***
dayThursday  -0.0953588  0.0251195  -3.796 0.000148 ***
dayTuesday   -0.1796350  0.0251832  -7.133 1.06e-12 ***
dayWednesday -0.1643719  0.0251080  -6.547 6.22e-11 ***
year2012      0.4983985  0.0135003  36.918  < 2e-16 ***
month02       0.1856071  0.0331645   5.597 2.25e-08 ***
month03       0.2969402  0.0353545   8.399  < 2e-16 ***
month04       0.5054469  0.0379837  13.307  < 2e-16 ***
month05       0.7386265  0.0427898  17.262  < 2e-16 ***
month06       0.6866161  0.0489667  14.022  < 2e-16 ***
month07       0.5557045  0.0549033  10.122  < 2e-16 ***
month08       0.6079919  0.0530521  11.460  < 2e-16 ***
month09       0.6925012  0.0472377  14.660  < 2e-16 ***
month10       0.8268814  0.0407215  20.306  < 2e-16 ***
month11       0.7652019  0.0349816  21.874  < 2e-16 ***
month12       0.7295816  0.0345119  21.140  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.619 on 8662 degrees of freedom
Multiple R-squared:  0.8293,	Adjusted R-squared:  0.8284 
F-statistic: 895.7 on 47 and 8662 DF,  p-value: < 2.2e-16
```

- Over all model was significant with p-value < 0.01
- Predictors weather, temp, humidity, windspeed, hour, day, year and month where significant with p-value<0.05
- There is a very high ***improvement in Adjusted R-Square value from 0.6911 to 0.8284***
- RMSE has gone very low from 101.2 to 0.619
- This, assure efficiency brought into this model using transformation, this model ***explains about 82.84% of variance*** in Count of Bicycles needed for users.

Residual Analysis on Transformed Model



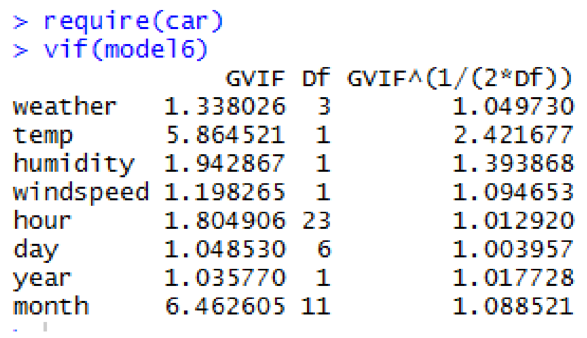
- From above residual graph we can infer that there I ***no more pattern exist*** which indicates no correlation with Residual and Fitted. This ***satisfies our assumption-1*** of error terms to be independent to each other

- From the same residual plot, we also see the ***funnel shape no more exists*** and hence can be proved that ***model is no more suffering from Heteroscedasticity***. Model is now Homoscedastic.

- S-shape in normal probability plot is also corrected to some extent which means our error terms are normally distributed. This ***satisfies our assumption-2*** of error being normally distributed.

- Check on outlier and Influential point was also done which again proved that they ***are natural outliers*** in the system and can be treated as it is in the data.

Multicollinearity Check on Transformed Model



```
> require(car)
> vif(model6)
              GVIF Df GVIF^(1/(2*Df))
weather   1.338026  3        1.049730
temp      5.864521  1        2.421677
humidity  1.942867  1        1.393868
windspeed 1.198265  1        1.094653
hour      1.804906 23        1.012920
day       1.048530  6        1.003957
year      1.035770  1        1.017728
month     6.462605 11        1.088521
```

- Building an efficient model and improving the Adjusted R-square value from 27.11% to 82.84% our final check will be with validating any multicollinearity in the final model.

- From above output we can infer that variance inflation factor (VIF) between all dependent variable is less than 10.

- This proves the ***absence of Multicollinearity in our transformed model***.

Model Validation:

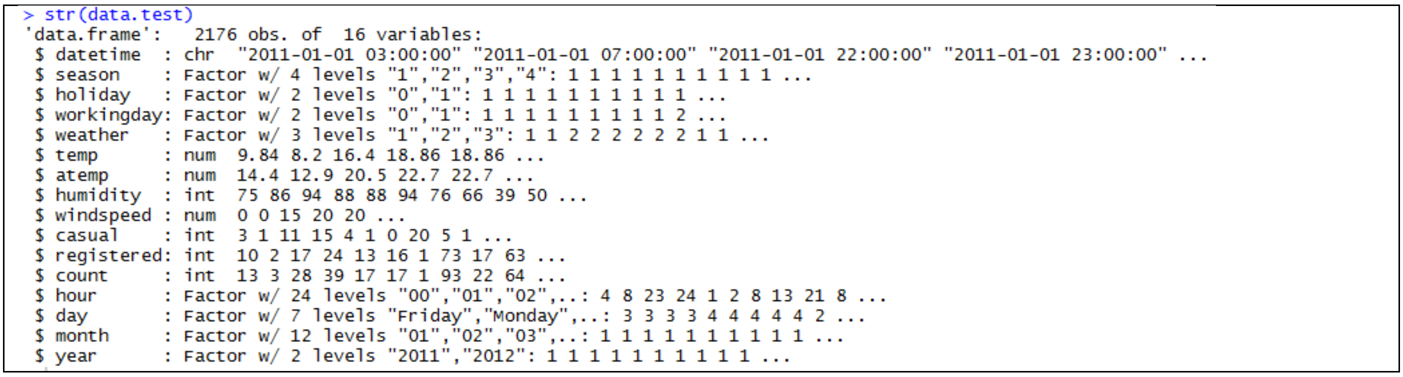
This is the final stage in building an analytical model. This validation will confirm the following:

1. Is the model over fitting?
2. Is the model under fitting?
3. Is the model good for all samples of the population?

In order to test our model log transformed model[M] we will be using the data.test which we had split as part of initial data split[S].

Preparing Test data

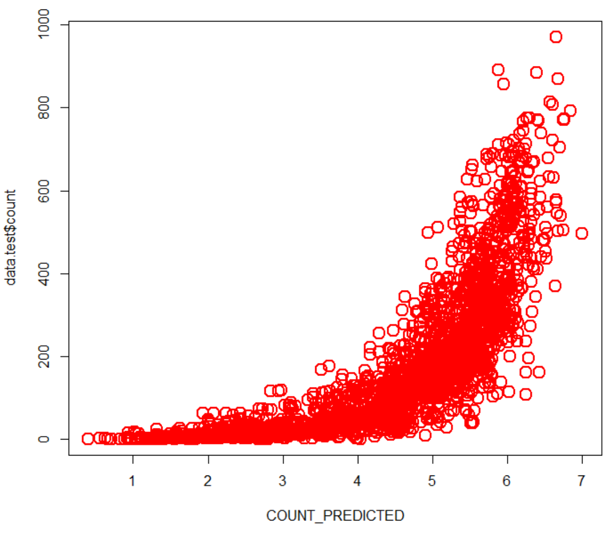
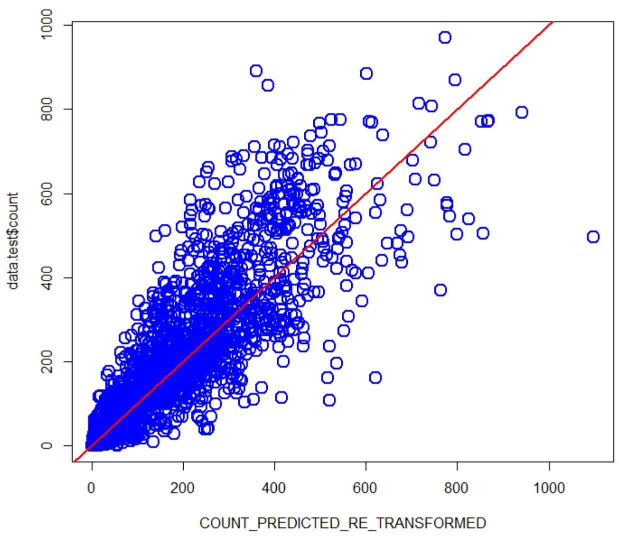
Our data.test was split from the train.csv file and hence they will be missing the extra features like weekdays, Month, Hours and Years that we had built as part of Binning[B]. So we will transform our data.test with necessary new features available by doing Binning on data.test again[R]. Post binning our data.test will look like below:



```
> str(data.test)
'data.frame':	2176 obs. of  16 variables:
 $ datetime  : chr  "2011-01-01 03:00:00" "2011-01-01 07:00:00" "2011-01-01 22:00:00" "2011-01-01 23:00:00" ...
 $ 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": 1 1 1 1 1 1 1 1 1 2 ...
 $ weather   : Factor w/ 3 levels "1","2","3": 1 1 2 2 2 2 2 2 1 1 ...
 $ temp      : num  9.84 8.2 16.4 18.86 18.86 ...
 $ atemp     : num  14.4 12.9 20.5 22.7 22.7 ...
 $ humidity  : int  75 86 94 88 88 94 76 66 39 50 ...
 $ windspeed : num  0 0 15 20 20 ...
 $ casual    : int  3 1 11 15 4 1 0 20 5 1 ...
 $ registered: int  10 2 17 24 13 16 1 73 17 63 ...
 $ count     : int  13 3 28 39 17 17 1 93 22 64 ...
 $ hour      : Factor w/ 24 levels "00","01","02",..: 4 8 23 24 1 2 8 13 21 8 ...
 $ day       : Factor w/ 7 levels "Friday","Monday",..: 3 3 3 3 4 4 4 4 4 2 ...
 $ month     : Factor w/ 12 levels "01","02","03",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ year      : Factor w/ 2 levels "2011","2012": 1 1 1 1 1 1 1 1 1 1 ...
```

Model Performance through Prediction

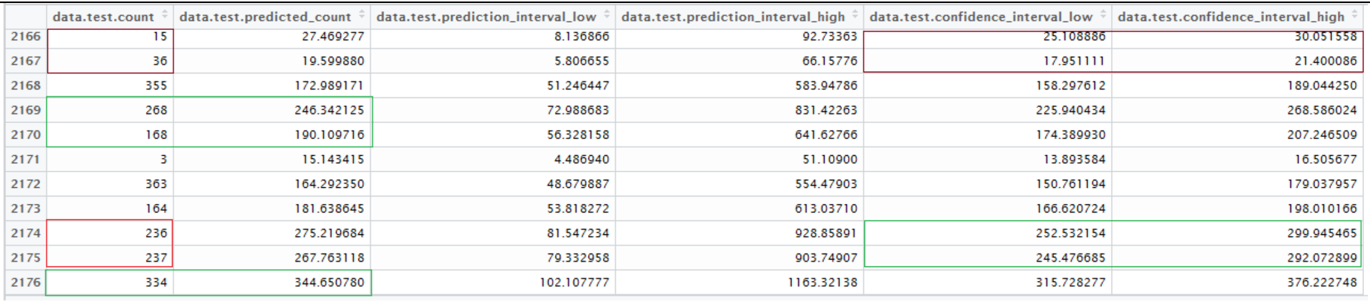
As part of this process we will inject our data.test into our log transformed model and look at its prediction. Since, we have done log transformation on dependent variable “count”, ***output from prediction needs to be subjected to exponent to get the final prediction value for count variable***. Below are the graphs of prediction performance of our model without exponentiation and with exponentiation.

<table>
<tr><th>Prediction Performance on Log Transformed Model</th><th>Prediction Performance on model after exponentiation</th></tr>
<tr><td>
</td><td>
</td></tr>
<tr><td colspan="2">➢ Above graph in red shows the model performance for the log transformed model
➢ While graph in blue shows the model performance for final model which was re-tansformed
➢ Final model performance proves that <b><i>our model is able to predict the value of count more efficiently</i></b> with no overfitting or underfitting. This proves the 82.44% of variance being explained by the final model proposed by us.</td></tr>
</table>

Model Prediction with Confidence and Prediction Interval values:

Below is the snapshot of model performance, it consists of below details:

1. Actual Count Value
2. Predicted Count Value
3. Predicted Interval Low
4. Predicted Interval High
5. Confidence Interval Low
6. Confidence Interval High



	data.test.count	data.test.predicted_count	data.test.prediction_interval_low	data.test.prediction_interval_high	data.test.confidence_interval_low	data.test.confidence_interval_high
2166	15	27.469277	8.136866	92.73363	25.108886	30.051558
2167	36	19.599880	5.806655	66.15776	17.951111	21.400086
2168	355	172.989171	51.246447	583.94786	158.297612	189.044250
2169	268	246.342125	72.988683	831.42263	225.940434	268.586024
2170	168	190.109716	56.328158	641.62766	174.389930	207.246509
2171	3	15.143415	4.486940	51.10900	13.893584	16.505677
2172	363	164.292350	48.679887	554.47903	150.761194	179.037957
2173	164	181.638645	53.818272	613.03710	166.620724	198.010166
2174	236	275.219684	81.547234	928.85891	252.532154	299.945465
2175	237	267.763118	79.332958	903.74907	245.476685	292.072899
2176	334	344.650780	102.107777	1163.32138	315.728277	376.222748

- ✓ From above image we could see how closely our model is able to predict the count values
- ✓ Box in green color highlights the ***closely matched values***
- ✓ Box in brown color highlights the count values falling into the closest confidence Interval range
- ✓ Box in red color highlights the count values falling into the closest prediction interval range
- ✓ This result proves us the efficacy of our model built

Final Proposed Model to predict the count of bikes needed to meet user demands:

Final Model to obtain a regression equation:

Log(count) ~ weather + temp + humidity + windspeed + hours + day + year + month

Final Regression Model:

Log(count) =

2.735
– (0.04)*weather2 – (0.55)*weather3
+ (0.02)*temp
-(0.002)*Humidity
-(0.003)*windspeed
-(0.65)*hour01-(1.21)*hour02-(1.78)*hour03-(2.04)*hour04-(0.99)*hour05
+(0.25)*hour06+(1.23)*hour07+(1.88)*hour08+(1.57)*hour09+(1.24)*hour10+(1.36)*hour11+(1.56)*hour12
+(1.53)*hour13+(1.44)*hour14+(1.5)*hour15+(1.77)*hour16+(2.17)*hour17+(2.1)*hour18+(1.79)*hour19
+(1.5)*hour20+(1.23)*hour21+(0.99)*hour22+(0.58)*hour23
-(0.15)*dayMonday – (0.094)*daySunday – (0.095)*dayThursday – (0.18)*dayTuesday – (0.16)*dayWednesday
+(0.5)*Year2012
+(0.18)*month02 + (0.29)*month03 + (0.5)*month04 + (0.73)*month05 + (0.68)*month06 + (0.55)*month07
+(0.6)*month08 + (0.69)*month09 + (0.82)*month10 + (0.76)*month11 + (0.72)*month12

Most Influential Variables and Inference:

- ✓ We computed standardized coefficient and derived at below results:
- ✓ In hours we see Hour08, Hour16, Hour17, Hour18, Hour19 as top positive influencing hours for bike needs
- ✓ Weather-3 (Snow/Rain) have a highest negative influence in bike need
- ✓ Demand is high on weekdays for Subscribed user and High on weekends for casual users
- ✓ Demand is based on current year and demands seems to be increases from 2011 to 2012
- ✓ May, June, Sep, Oct, Nov and Dec are months with highest demand

Timetable

Phases	Description of Work	Start and End Dates
Phase One	Obtaining Dataset from Kaggle	18-Oct to 24-Oct 2016
Phase Two	Performing EDA on Bike Sharing Dataset	25-Oct to 31-Oct-2016
Phase Three	Basic Model Building	01-Nov to 07-Nov-2016
Phase Four	Variable Binning	01-Nov to 07-Nov-2016
Phase Five	Transformations	08-Nov to 14-Nov-2016
Phase Six	Trying advance model building (SVM, DT)	08-Nov to 14-Nov-2016
Phase Seven	Testing and Validation	15-Nov to 20-Nov-2016
Phase Eight	Report Writing and Review	15-Nov to 20-Nov-2016
Phase Nine	Deliverable Submission	21-Nov-2016

Key Personnel

Team Member	Pradeep Sathyamurthy
Professor	Prof. Nandhini Gulasingam
Project for	CSC-423
Target Team	DePaul CDM

Deliverables

Final Report	**Prady_CSC_423_Technical_Report.pdf**	Contains final Technical Report
Final Report	**Prady_CSC_423_Non_Technical_Report.pdf**	Contains final non-technical report
Raw Data Set	**train.csv**	Raw Dataset downloaded from Kaggle
R	**Prady_Source_Files_Bike_Share.R**	Source File to Run through
R_Data_Files	**Prady_Split_80_20_Train_Test_Dataset.RData**	Can be loaded in R to test the code
R_Data_Files	**Prady_Project_All_Outcomes.RData**	Can be loaded in R to test all o/p

Reference

[1] Bike Share Dataset in Kaggle Website *https://www.kaggle.com/c/bike-sharing-demand*

[2] Publicly available capital bike shares program data *http://capitalbikeshare.com/system-data*

[3] Data from which weather data was filled *http://www.freemeteo.com*

[4] Citation Request: Fanaee-T, Hadi, and Gama, Joao, Event labelling combining ensemble detectors and background knowledge*, Progress in Artificial Intelligence (2013): pp. 1-15, Springer Berlin Heidelberg.*

[5] Comparing my model with others, *http://rstudio-pubs-static.s3.amazonaws.com/25024_ab1590e0d42d4443b88d30b9baf86897.html*

Zip Files

Prady_CSC_423_Deliverables.rar

Appendix

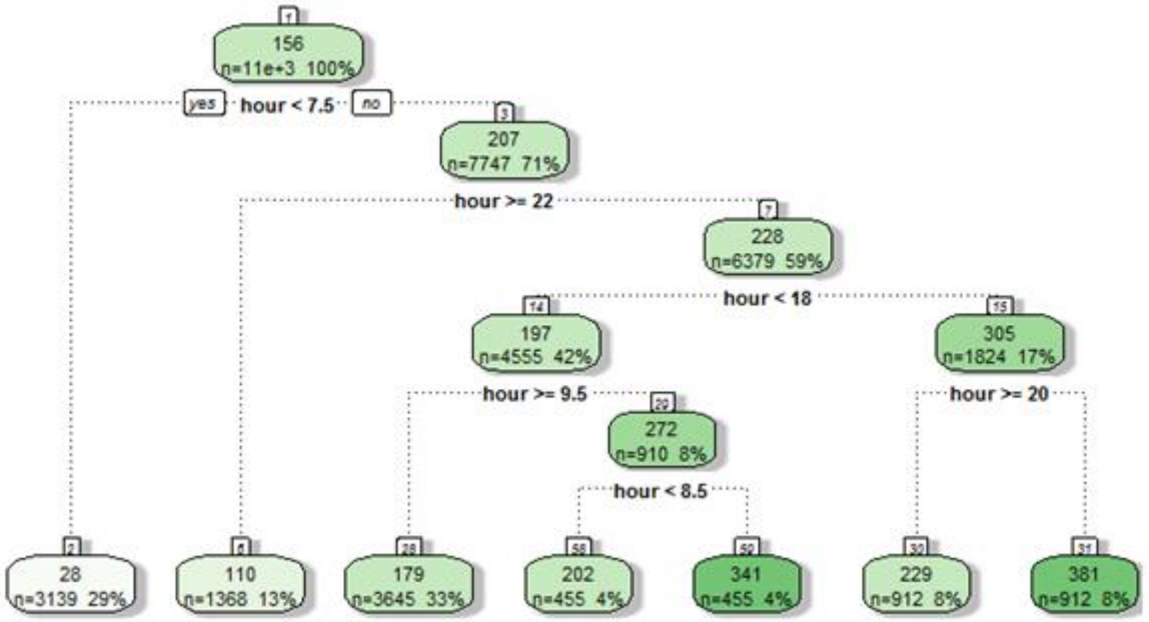
Scope for improvement

I tried to contact few of the data scientist from who has high score for this dataset submission to find a way for improvement. Below are few suggestions given:

1. They suggested ***my approach will be in top 10 percentile of submission*** by providing a reference[5] URL and saying my model explains more variance than that.

2. They also suggested me to use ***decision tree*** to cluster my "Hours", "Months" and "weekdays" variable which can ***reduce the number of dummy variables*** in the model and has a good scope for model improvement.

3. Few suggested me to use ***Support Vector Machine*** and few suggested me to use gradient boost algorithms after grouping as said above which will further increase the model efficiency.



Rattle 2016-Nov-15 14:31:33 prade

- ✓ I ***tried with decision tree*** and able to do clustering, I will further work and see how LR model help by that grouping.

- ✓ I was able to ***reduce the features of Hours from 23 variables to 8 clusters***.

- ✓ I replaced these new grouped 8 variables in place of 23 Hours dummy variables, ***I was able to get an adjusted R-square value of 89.45%*** I would like to use SVM and see what is the maximum variance I can achieve with this model.

R-Code for Bike Share Dataset

```
###############################################################################################
# Author: Pradeep Sathyamurthy, Daniel Glownia
# Guiding Professor: Prof. Nandhini Gulasingam
# Course: CSC-423
# Project: Final Course Project for CSC-423, visualizing Kaggle’s Bike Share dataset
# Part-1: Building simple linear regression model
# Part-2: Variable Binning and then building a linear regression model
# Part-3: Performed Log Transformation on Dependent variable for model improvement
# Date Created: 21-Oct-2016
# Date Last Modified: 23-Nov-2016
###############################################################################################

# Analyzing the dataset
getwd()
setwd("D:/Courses/CSC423 - SAS & R - Data Analysis and Regression/SAS/Project")
list.files(getwd())

# Importing the dataset
data.kaggle <- read.csv("train.csv",na.strings="Not Available",stringsAsFactors=FALSE)
head(data.kaggle,3)

# Variable type identification and finding missing values
str(data.kaggle)
table(is.na(data.kaggle))

# EDA on dataset
par(mfrow=c(4,2))
par(mar = rep(2, 4))
hist(data.kaggle$season)
hist(data.kaggle$weather)
hist(data.kaggle$humidity)
hist(data.kaggle$holiday)
hist(data.kaggle$workingday)
hist(data.kaggle$temp)
hist(data.kaggle$atemp)
hist(data.kaggle$windspeed)
prop.table(table(data.kaggle$weather))

#########################################
# Data Splitting (Train:Test = 80:20)
#########################################

# Installing the necessary library package
require(caret)
# Creating a random index to split the data as 80 - 20%
idx <- createDataPartition(data.kaggle$count, p=.80, list=FALSE)
#print(idx)
# Using the index created to create a Training Data set - 131 observations created
data.train <- data.kaggle[idx,]
head(data.train)
# Using the index created to create a Testing Data set - 31 observations created
data.test <- data.kaggle[-idx,]
head(data.test)
idx <- NULL
```

```
##############################################
# Dummy Variable or Factor variable creation
##############################################

table(data.train$season) # has 4 levels, thus need of 3 dummy variables
table(data.train$holiday) # this is just a binary variable
table(data.train$workingday) # this is again a binary variable
table(data.train$weather) # has 4 levels, thus need of 3 dummy variables

str(data.train)
data.train$season <- as.factor(data.train$season)
data.train$holiday <- as.factor(data.train$holiday)
data.train$workingday <- as.factor(data.train$workingday)
data.train$weather <- as.factor(data.train$weather)

#############################################################################
# Model Building - Ignoring datetime variable and using all other variable
#############################################################################

model1_count <-
lm(count~season+holiday+workingday+weather+temp+atemp+humidity+windspeed+casual+registered,
data=data.train)
summary(model1_count) # This gives Adj-R2 = 1, this is because variable casual + registred can calculate the count
perfectly
# So features casual and registered are part of dependent variable and should be excluded from model
model2_count <- lm(count~season+holiday+workingday+weather+temp+atemp+humidity+windspeed,
data=data.train)
summary(model2_count)
# model equation
# count = 115.3123 - 39.2224*(season3) + 64.9691*(season4) + 13.5484*(weather2) + 7.7457*(temp) +
3.0836*(atemp) -2.7415*(humidity) + 0.6616*(windspeed)
# However, adj R^2 is 0.2716

plot(model2_count)
df.temp <- data.frame(data.train$temp,data.train$atemp,data.train$humidity,data.train$windspeed)
cor(df.temp)
#                   data.train.temp data.train.atemp data.train.humidity data.train.windspeed
#data.train.temp         1.00000000      0.98519193       -0.06857852         -0.02005605
#data.train.atemp        0.98519193      1.00000000       -0.04880235         -0.05791124
#data.train.humidity    -0.06857852     -0.04880235        1.00000000         -0.31812732
#data.train.windspeed   -0.02005605     -0.05791124       -0.31812732          1.00000000
# we clearly see a strong corelation between temp and atemp
# building a model by removing atemp
require(car)
vif(model2_count)
model3_count <- lm(count~season+holiday+workingday+weather+temp+humidity+windspeed, data=data.train)
summary(model3_count)
# Adj-R^2 is still 0.2711

# we will try to deploy even variable selction method and try to build a optimum model
model3_count_stepwise <- step(model3_count, direction = "both")
summary(model3_count_stepwise)
# count = 120.2626 - 41.5517*(season3) + 65.3086*(season4) + 13.3975*(weather2) - 13.3334*(weather3) +
11.1348*(temp) - 2.7235*(humidity) + 0.5443*(windspeed)
# However, adj R^2 is 0.2712
```

```
###########################################################################################
# Bringing datetime variable into consideration for model building - Feature Engineering
###########################################################################################

str(data.train)
# we see datatime feature is of type "chr" with sample value like "2011-01-01 01:00:00"
# let us try to split this into Years, Month, weekdays and hours

# Subsetting hours from a datetime field
data.train$hour <- substr(data.train$datetime,12,13)
data.train$hour<- as.factor(data.train$hour)

# Subsetting weekday
date <- substr(data.train$datetime,1,10)
# Creating days from date
data.train$day <- weekdays(as.Date(date))
data.train$day <- as.factor(data.train$day)

# Seperating Month from the date
data.train$month=substr(data.train$datetime,6,7)
data.train$month=as.factor(data.train$month)

# Seperating Years from the date
data.train$year=substr(data.train$datetime,1,4)
data.train$year=as.factor(data.train$year)


###############################################################################################
#####
# Model Building - Including variables created from datetime that is hour, weekday, Month and Year
###############################################################################################
#####

model4_count                                                                                    <-
lm(count~season+holiday+workingday+weather+temp+humidity+windspeed+hour+day+year+month,
data=data.train)
#summary(model4_count)
model4_count_stepwise <- step(model4_count,direction = "both")
summary(model4_count_stepwise)
# This increases adj R^2 to 0.6911

model4_count_backward <- step(model4_count,direction = "backward")
summary(model4_count_backward)

model5 <- lm(count~weather+temp+humidity+windspeed+hour+day+year+month, data = data.train)
summary(model5)
sqrt(anova(model5))
# Adj R^2 is 0.6911

# Plotting model
par(mfrow=c(4,2))
par(mar = rep(2, 4))
plot(data.train$temp,data.train$count)
plot(data.train$humidity,data.train$count)
plot(data.train$windspeed,data.train$count)

# Interaction Model
```

```
model5_interaction <- lm(count~weather+temp*humidity*windspeed+hour+day+year+month, data = data.train)
summary(model5_interaction)
sqrt(anova(model5_interaction))

# Polynomial Model
model5_polynomial <-
lm(count~weather+temp*temp+humidity*humidity+windspeed*windspeed+hour+day+year+month, data =
data.train)
summary(model5)
sqrt(anova(model5_polynomial))

###############################################################################################
# Residual Analysis
###############################################################################################
par(mfrow=c(4,2))
par(mar = rep(2, 4))
plot(model5)
sd(data.train$count)
residual <- rstandard(model5)
hist(residual)
# Ensures presence of heteroscadasity, so lets try with transformation

###############################################################################################
# Log Transformation
###############################################################################################
# Doing log transformation on dependent variable
model6 <- lm(log(count)~weather+temp+humidity+windspeed+hour+day+year+month, data = data.train)
summary(model6)
# Adj R^2 is 0.8284
# Amazing, great improvemnt
par(mfrow=c(4,2))
par(mar = rep(2, 4))
plot(model6)

# Trying with few interaction terms
model7 <- lm(log(count)~weather+temp*humidity*windspeed+hour+day+year+month, data = data.train)
summary(model7)
# Adj R^2 is 0.8288, this is not a significant improvement over the ordinary linear model, so we can drop it

###############################################################################################
# OUtlier and Influential Point Check
###############################################################################################

# computing studentized residual for outlier check
require(MASS)
n_sample_size <- nrow(data.train)
studentized.residuals <- studres(model6)
#cat("Complete list of Studentized Residual::::","\n")
#print(studentized.residuals)
for(i in c(1:n_sample_size)){
   if(studentized.residuals[i] < -3 || studentized.residuals[i] > 3){
      cat("Validate these values for outliers:::",studentized.residuals[i],"at observation",i,"\n")
   }
}

# Influential Points
hhat.model <- lm.influence(model6)$hat
```

```
n_sample_size <- nrow(data.train)
p_beta <- length(model6$coefficients) +1
#cat("Complete list of HHat Values::::","\n")
#print(hhat.model)
hhat.cutoff <- (2*p_beta)/n_sample_size
cat("Looking for values more than cut off::::",hhat.cutoff,"\n")
for(i in c(1:n_sample_size)){
  if(hhat.model[i] > hhat.cutoff){
    cat("Validate these values for Influential points:::",hhat.model[i],"at observation",i,"\n")
  }
}
# None of the outliers are part of influential points, this reconfirms these are natural outliers
# Hence, we cannot remove the same

#########################################################################################
# Reconfirming the absence of multicollinearity
#########################################################################################
# Checking Variance Inflation Factor
require(car)
vif(model6)

#########################################################################################
# Computing the standardized coefficients
#########################################################################################
data.train.std <- sapply(data.train[,],FUN=scale)
data.train.std <- data.frame(data.train.std)
model6.final.std <- lm(log(count)~weather+temp*humidity*windspeed+hour+day+year+month, data = data.train)
summary(model6.final.std)


####################
#Model Validation
####################

FINAL_MODEL <- model6
final_summary <- summary(FINAL_MODEL); final_summary

# Feature engineering on Test Data
data.test$season <- as.factor(data.test$season)
data.test$holiday <- as.factor(data.test$holiday)
data.test$workingday <- as.factor(data.test$workingday)
data.test$weather <- as.factor(data.test$weather)
data.test$hour <- substr(data.test$datetime,12,13)
data.test$hour<- as.factor(data.test$hour)
date <- substr(data.test$datetime,1,10)
data.test$day <- weekdays(as.Date(date))
data.test$day <- as.factor(data.test$day)
data.test$month=substr(data.test$datetime,6,7)
data.test$month=as.factor(data.test$month)
data.test$year=substr(data.test$datetime,1,4)
data.test$year=as.factor(data.test$year)
str(data.test)

#Prediction
COUNT_PREDICTED <- predict(FINAL_MODEL,data.test)
plot(COUNT_PREDICTED,data.test$count,lwd=2, cex=2, col="red")
COUNT_PREDICTED_RE_TRANSFORMED <- exp(COUNT_PREDICTED)
```

```
plot(COUNT_PREDICTED_RE_TRANSFORMED,data.test$count,lwd=2, cex=2, col="green")
abline(0,1,col='red', lwd=2)

# Prediction Interval
pred_Int <- predict(FINAL_MODEL,data.test,interval = "predict")
conf_Int <- predict(FINAL_MODEL,data.test,interval = "confidence")
converted_pred_int <- exp(pred_Int)
converted_conf_int <- exp(conf_Int)
data.test$predicted_count <- converted_pred_int[,1]
data.test$prediction_interval_low <- converted_pred_int[,2]
data.test$prediction_interval_high <- converted_pred_int[,3]
data.test$confidence_interval_low <- converted_conf_int[,2]
data.test$confidence_interval_high <- converted_conf_int[,3]
data.prediction.result <- data.frame(data.test$count,data.test$predicted_count,data.test$prediction_interval_low,data.test$prediction_interval_high,data.test$confidence_interval_low,data.test$confidence_interval_high)
View(data.prediction.result)
data.test$predicted_count <- NULL
data.test$prediction_interval_low <- NULL
data.test$prediction_interval_high <- NULL
data.test$confidence_interval_low <- NULL
data.test$confidence_interval_high <- NULL
```