GROUP - 11

BIKE RENTAL COUNT PREDICTION : STASTICAL TECHNIQUES

Advanced Statistical Learning - II

INSTRUCTOR : CHING CHI YANG

ABHISHEK BOGA - U00884042

VISHNU CHOUNDUR - U00869233

MANASWINI KOMMAREDDY - U00881454

JOHNPAUL CHIEMELIE ANAMEGE - U00870359

SAI NARENDRA REDDY LAKKIREDDY - U00892852

CONTENTS

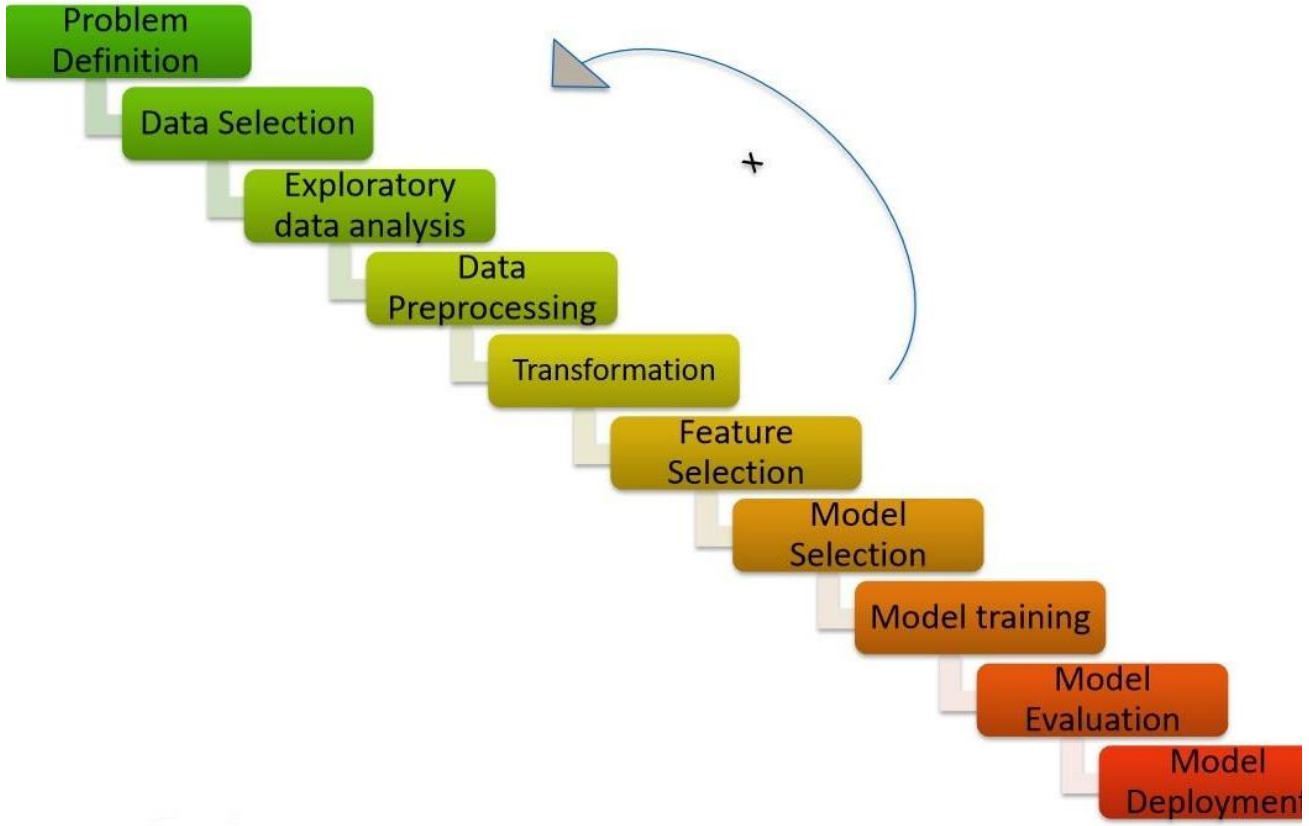
INTRODUCTION

- At the heart of our project lies a singular goal: to craft a predictive model that not only reads the patterns of bike usage but also predicts them with precision. To this end, we are rigorously testing a spectrum of machine learning models, engaging in a comparative analysis to determine which algorithm reigns supreme in forecasting bike count based on variables like working days, seasons, and other vital parameters
- Beyond the aim of prediction accuracy, our mission is to meticulously evaluate and identify the most effective model from a cadre of contenders.
- Through this exploration, we aim to distill the essence of what makes a model not just functional, but exceptional, in predicting bike-sharing demand.

MACHINE LEARNING LIFE CYCLE

- **Problem Statement** : To predict the bike rental count based on hourly based on the environmental and seasonal settings.
- **Data Selection** :
  - Source : UC Irvine Machine Learning Repository,
  - Link : https://archive.ics.uci.edu/dataset/275/bike+sharing+dataset
  - Here, two dataset we have : 1. hour.csv (17379*17) and day.csv (731*16).
  - We choose hour.csv dataset.
  - Dimension dataset : 17379 *17
  - 1 – Index, 2 – date, 3-10 – Categorical,

    11-14 – Numerical and 15 – 17 – Dependent Variables



Data Set

instant	dteday	season	yr	mnth	hr	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	casual	registered	cnt
1	2011-01-01	1	0	1	0	0	6	0	1	0.24	0.2879	0.81	0.0000	3	13	16
2	2011-01-01	1	0	1	1	0	6	0	1	0.22	0.2727	0.80	0.0000	8	32	40
3	2011-01-01	1	0	1	2	0	6	0	1	0.22	0.2727	0.80	0.0000	5	27	32
4	2011-01-01	1	0	1	3	0	6	0	1	0.24	0.2879	0.75	0.0000	3	10	13
5	2011-01-01	1	0	1	4	0	6	0	1	0.24	0.2879	0.75	0.0000	0	1	1
6	2011-01-01	1	0	1	5	0	6	0	2	0.24	0.2576	0.75	0.0896	0	1	1
7	2011-01-01	1	0	1	6	0	6	0	1	0.22	0.2727	0.80	0.0000	2	0	2

- ✓ instant: record index
- ✓ dteday : date
- ✓ season : season (1:springer, 2:summer, 3:fall, 4:winter)
- ✓ yr : year (0: 2011, 1:2012)
- ✓ mnth : month (1 to 12)
- ✓ hr : hour (0 to 23)
- ✓ holiday : weather day is holiday or not
- ✓ weekday : day of the week
- ✓ workingday : if day is neither weekend nor holiday is 1, otherwise is 0.
- ✓ weathersit :
  - ➢ 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

- ✓ temp : Normalized temperature in Celsius. The values are divided to 41 (max)
- ✓ atemp: Normalized feeling temperature in Celsius. The values are divided to 50 (max)
- ✓ hum: Normalized humidity. The values are divided to 100 (max)windspeed: Normalized wind speed. The values are divided to 67 (max)
- ✓ casual: count of casual users
- ✓ registered: count of registered users
- ✓ cnt: count of total rental bikes including both casual and registered

```
> summary(data)
    instant          dteday              season           yr              mnth             hr             holiday       
 Min.   :    1   Length:17379       Min.   :1.000   Min.   :0.0000   Min.   : 1.000   Min.   : 0.00   Min.   :0.00000  
 1st Qu.: 4346   Class :character   1st Qu.:2.000   1st Qu.:0.0000   1st Qu.: 4.000   1st Qu.: 6.00   1st Qu.:0.00000  
 Median : 8690   Mode  :character   Median :3.000   Median :1.0000   Median : 7.000   Median :12.00   Median :0.00000  
 Mean   : 8690                      Mean   :2.502   Mean   :0.5026   Mean   : 6.538   Mean   :11.55   Mean   :0.02877  
 3rd Qu.:13034                      3rd Qu.:3.000   3rd Qu.:1.0000   3rd Qu.:10.000   3rd Qu.:18.00   3rd Qu.:0.00000  
 Max.   :17379                      Max.   :4.000   Max.   :1.0000   Max.   :12.000   Max.   :23.00   Max.   :1.00000  
    weekday        workingday      weathersit         temp            atemp              hum          windspeed      
 Min.   :0.000   Min.   :0.0000   Min.   :1.000   Min.   :0.020   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000  
 1st Qu.:1.000   1st Qu.:0.0000   1st Qu.:1.000   1st Qu.:0.340   1st Qu.:0.3333   1st Qu.:0.4800   1st Qu.:0.1045  
 Median :3.000   Median :1.0000   Median :1.000   Median :0.500   Median :0.4848   Median :0.6300   Median :0.1940  
 Mean   :3.004   Mean   :0.6827   Mean   :1.425   Mean   :0.497   Mean   :0.4758   Mean   :0.6272   Mean   :0.1901  
 3rd Qu.:5.000   3rd Qu.:1.0000   3rd Qu.:2.000   3rd Qu.:0.660   3rd Qu.:0.6212   3rd Qu.:0.7800   3rd Qu.:0.2537  
 Max.   :6.000   Max.   :1.0000   Max.   :4.000   Max.   :1.000   Max.   :1.0000   Max.   :1.0000   Max.   :0.8507  
     casual        registered         cnt        
 Min.   :  0.00   Min.   :  0.0   Min.   :  1.0  
 1st Qu.:  4.00   1st Qu.: 34.0   1st Qu.: 40.0  
 Median : 17.00   Median :115.0   Median :142.0  
 Mean   : 35.68   Mean   :153.8   Mean   :189.5  
 3rd Qu.: 48.00   3rd Qu.:220.0   3rd Qu.:281.0  
 Max.   :367.00   Max.   :886.0   Max.   :977.0  
```

```
> dim(data)   # 17379,17
[1] 17379    17
```

```
> str(data)
'data.frame':   17379 obs. of  17 variables:
 $ instant   : int  1 2 3 4 5 6 7 8 9 10 ...
 $ dteday    : chr  "2011-01-01" "2011-01-01" "2011-01-01" "2011-01-01" ...
 $ season    : int  1 1 1 1 1 1 1 1 1 1 ...
 $ yr        : int  0 0 0 0 0 0 0 0 0 0 ...
 $ mnth      : int  1 1 1 1 1 1 1 1 1 1 ...
 $ hr        : int  0 1 2 3 4 5 6 7 8 9 ...
 $ holiday   : int  0 0 0 0 0 0 0 0 0 0 ...
 $ weekday   : int  6 6 6 6 6 6 6 6 6 6 ...
 $ workingday: int  0 0 0 0 0 0 0 0 0 0 ...
 $ weathersit: int  1 1 1 1 1 2 1 1 1 1 ...
 $ temp      : num  0.24 0.22 0.22 0.24 0.24 0.24 0.22 0.2 0.24 0.32 ...
 $ atemp     : num  0.288 0.273 0.273 0.288 0.288 ...
 $ hum       : num  0.81 0.8 0.8 0.75 0.75 0.75 0.8 0.86 0.75 0.76 ...
 $ windspeed : num  0 0 0 0 0 0.0896 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 ...
 $ cnt       : int  16 40 32 13 1 1 2 3 8 14 ...
```

```
> for (col_name in names(data)) {
+   if (length(unique(data[[col_name]])) < 25) {
+     cat(paste("Unique ", col_name,"'s count: ", length(unique(data[[col_name]])), "\n"))
+     cat(paste("Unique values: ", paste(unique(data[[col_name]]), collapse = ", "), "\n\n"))
+   }
+ }
Unique  season 's count:  4
Unique values:  1, 2, 3, 4

Unique  yr 's count:  2
Unique values:  0, 1

Unique  mnth 's count:  12
Unique values:  1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12

Unique  hr 's count:  24
Unique values:  0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23

Unique  holiday 's count:  2
Unique values:  0, 1

Unique  weekday 's count:  7
Unique values:  6, 0, 1, 2, 3, 4, 5

Unique  workingday 's count:  2
Unique values:  0, 1

Unique  weathersit 's count:  4
Unique values:  1, 2, 3, 4
```

Data Preprocessing

- Handling Missing Data
- Data Cleaning
- Handling Date and Time Data

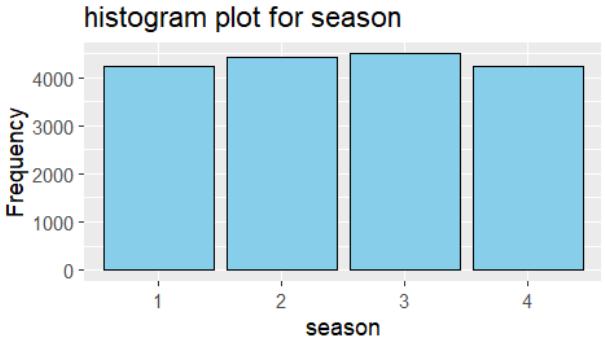
```
> #####
> ### Checking relationship between the dteday and yr
> data$Year <- format(as.Date(data$dteday),"%Y")
> data$Category <- ifelse(data$Year == "2011", 0, 1)
> identical_columns <- identical(data$yr, as.integer(data$Category))
> print(identical_columns)
[1] TRUE
> ### Checking relationship between the dteday and mnt
> data$month <- format(as.Date(data$dteday),"%m")
> identical_columns <- identical(data$mnth, as.integer(data$month))
> print(identical_columns)
[1] TRUE
> ### Checking relationship between the casual, registered and cnt
> data$cnt1 <- data$casual+data$registered
> identical_columns <- identical(data$cnt, data$casual+data$registered)
> print(identical_columns)
[1] TRUE
>
```

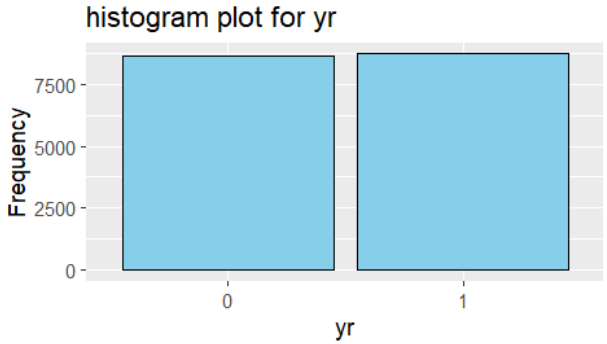
```
> ## Duplicate Data
> # Find and display duplicate rows based on all columns
> duplicate_rows <- data[duplicated(data) | duplicated(data, fromLast = TRUE), ]
> # Display the duplicate rows
> print("Duplicate Rows in the Dataset:")
[1] "Duplicate Rows in the Dataset:"
> print(duplicate_rows)
      season yr mnth hr holiday weekday workingday weathersit temp  atemp  hum windspeed cnt
7959       4  0   12  6       0       6          0          1 0.24 0.2576 0.65    0.1045  11
8127       4  0   12  6       0       6          0          1 0.24 0.2576 0.65    0.1045  11
13560      3  1    7  4       0       2          1          1 0.66 0.6061 0.83    0.0896   6
13728      3  1    7  4       0       2          1          1 0.66 0.6061 0.83    0.0896   6
```

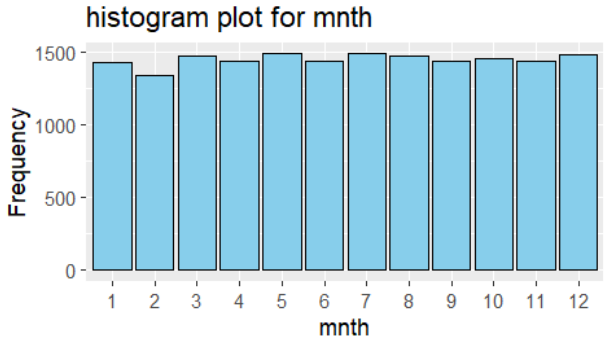
```
> print(missing_counts)
    season         yr       mnth         hr    holiday    weekday workingday weathersit       temp      atemp
         0          0          0          0          0          0          0          0          0          0
       hum  windspeed        cnt
         0          0          0
```

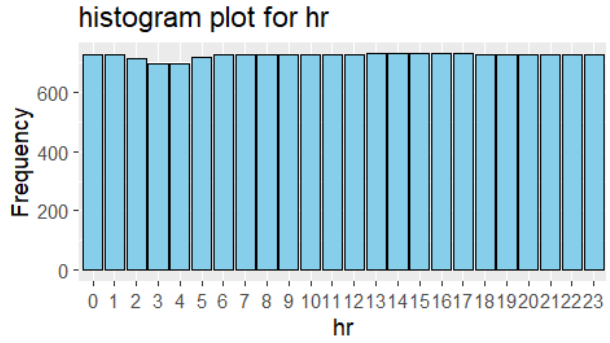
Data Visualization And Analysis

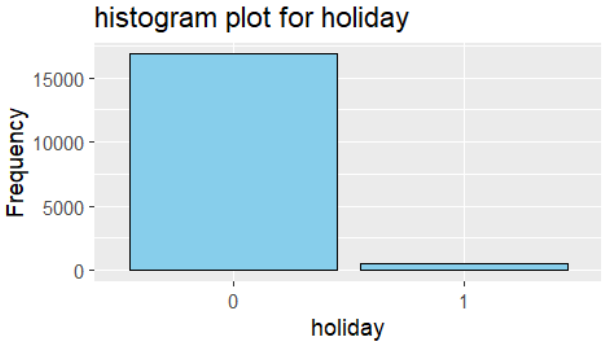
- Histogram
- Scatter Plot
- Box Plot
- Correlation

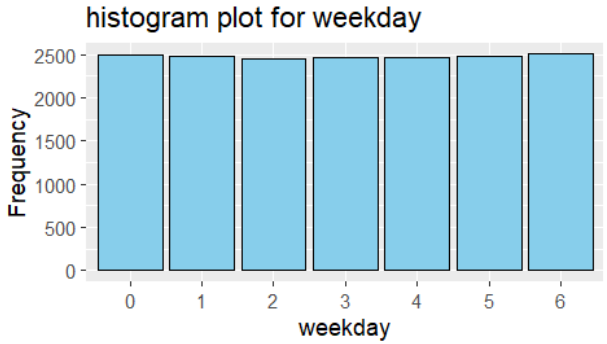

histogram plot for season
Frequency
season

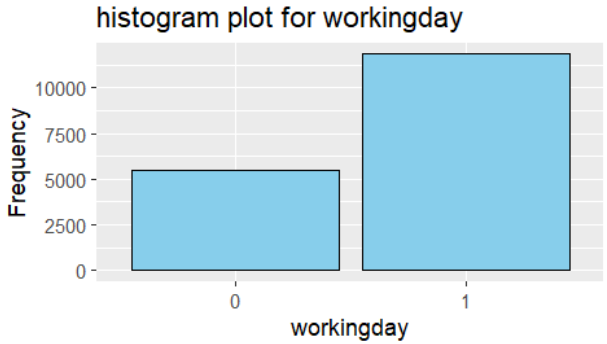

histogram plot for yr
Frequency
yr

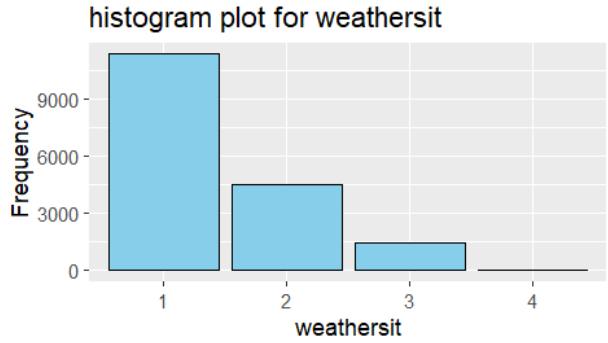

histogram plot for mnth
Frequency
mnth

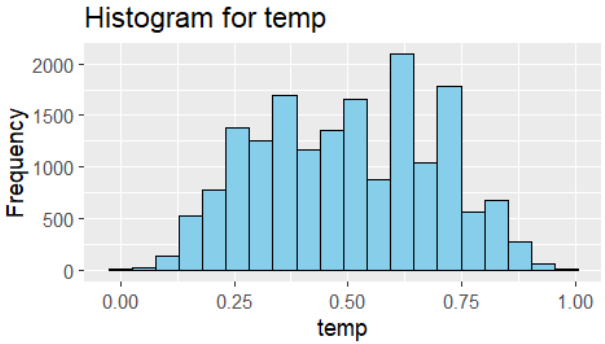

histogram plot for hr
Frequency
hr

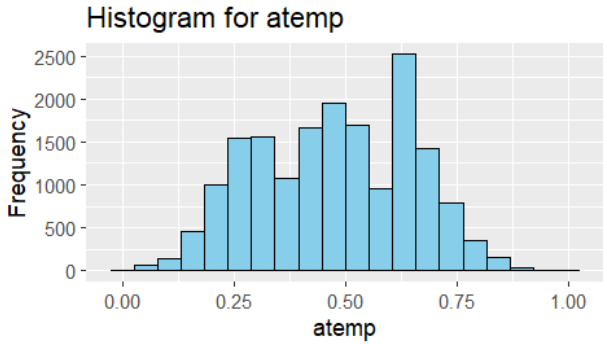

histogram plot for holiday
Frequency
holiday

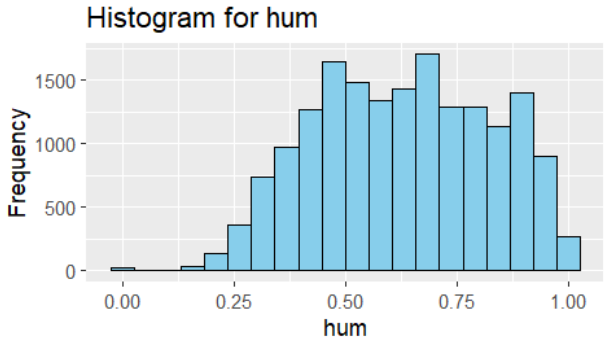

histogram plot for weekday
Frequency
weekday

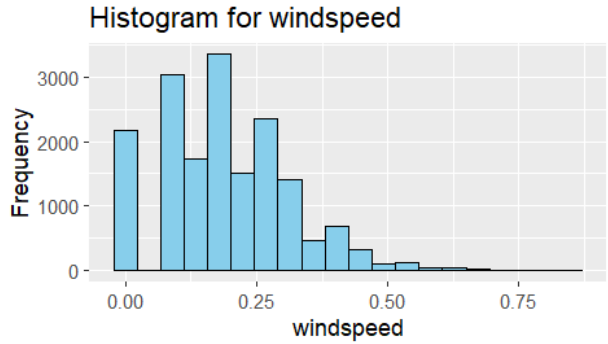

histogram plot for workingday
Frequency
workingday

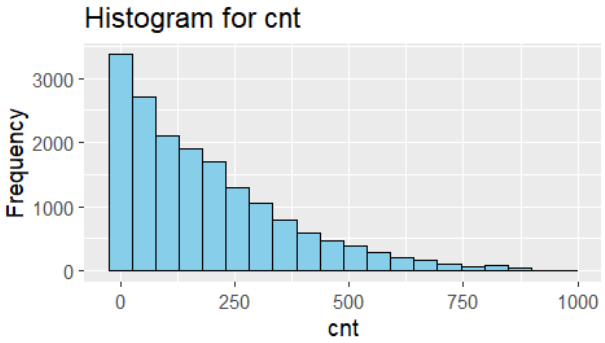

histogram plot for weathersit
Frequency
weathersit


Histogram for temp
Frequency
temp

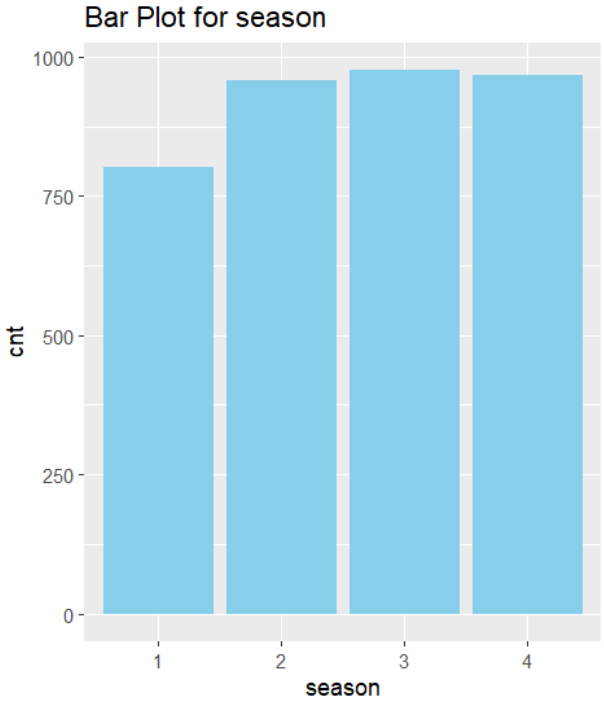

Histogram for atemp
Frequency
atemp

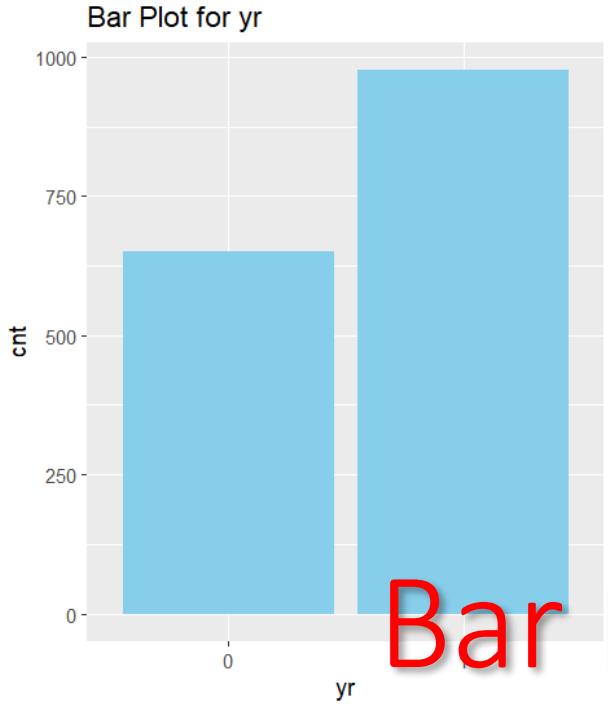

Histogram for hum
Frequency
hum

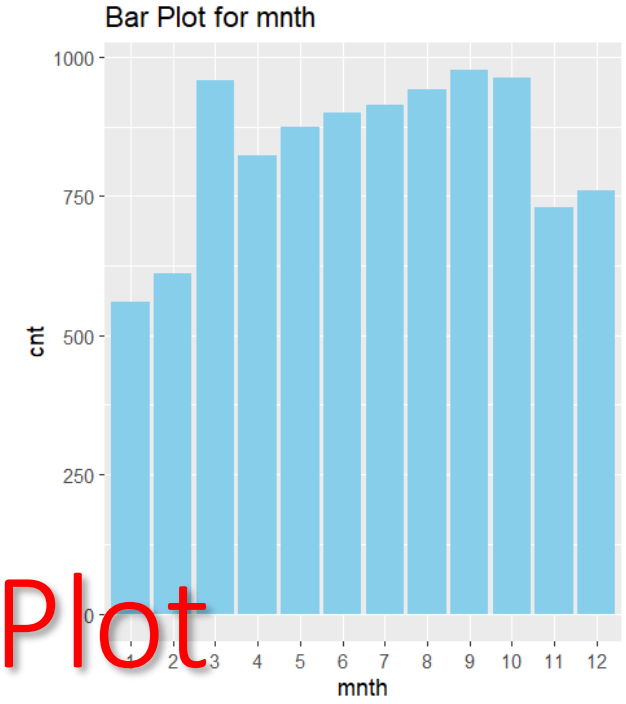

Histogram for windspeed
Frequency
windspeed

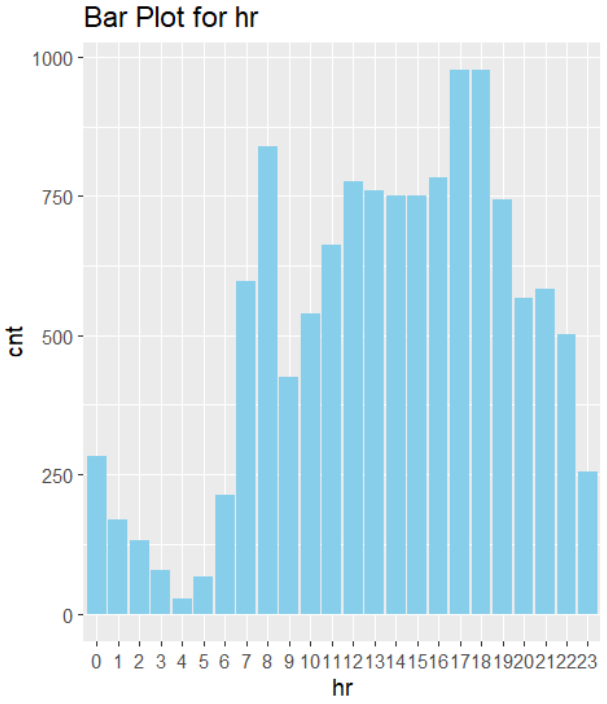

Histogram for cnt
Frequency
cnt

Histogram

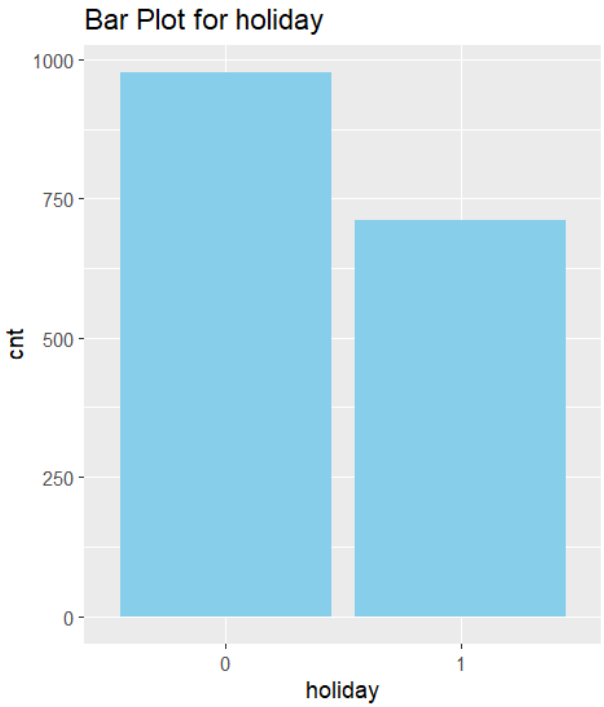

Bar Plot for season
cnt
1000
750
500
250
0
1
2
3
4
season

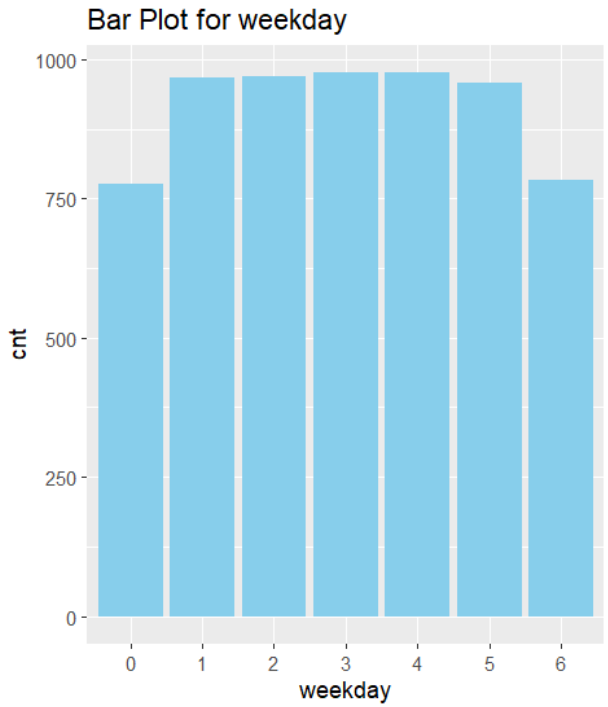

Bar Plot for yr
cnt
1000
750
500
250
0
0
yr

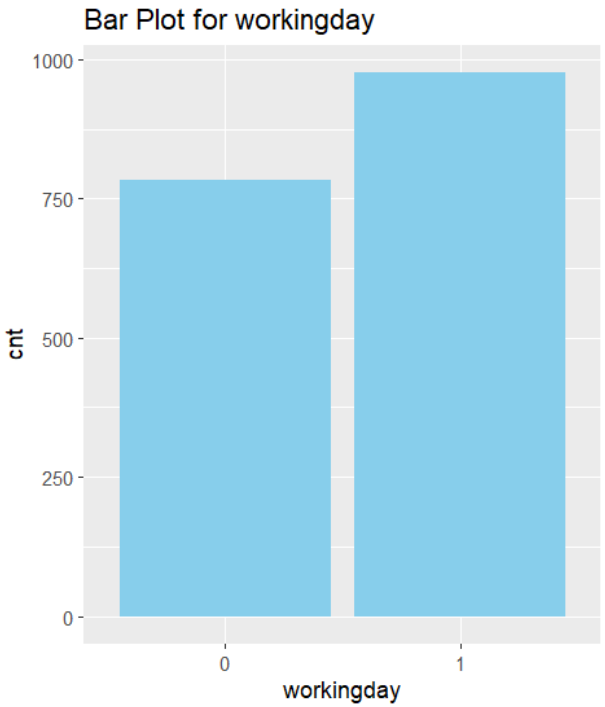

Bar Plot for mnth
cnt
1000
750
500
250
0
1 2 3 4 5 6 7 8 9 10 11 12
mnth

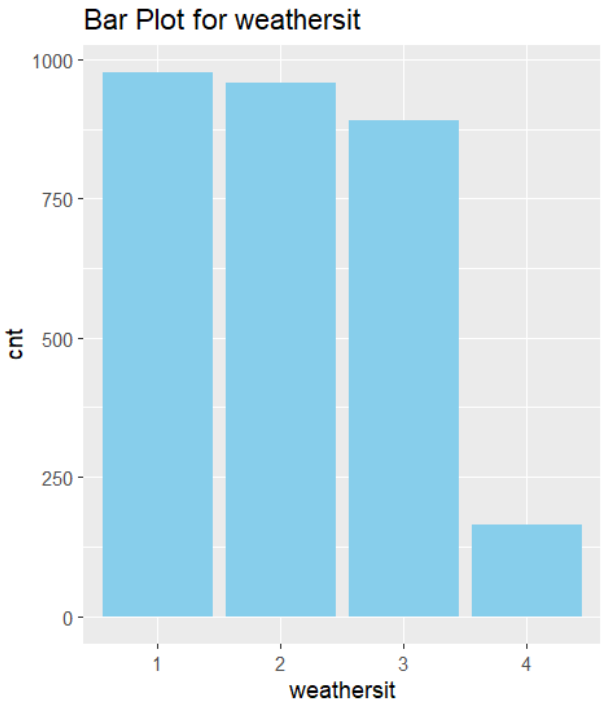

Bar Plot for hr
cnt
1000
750
500
250
0
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
hr

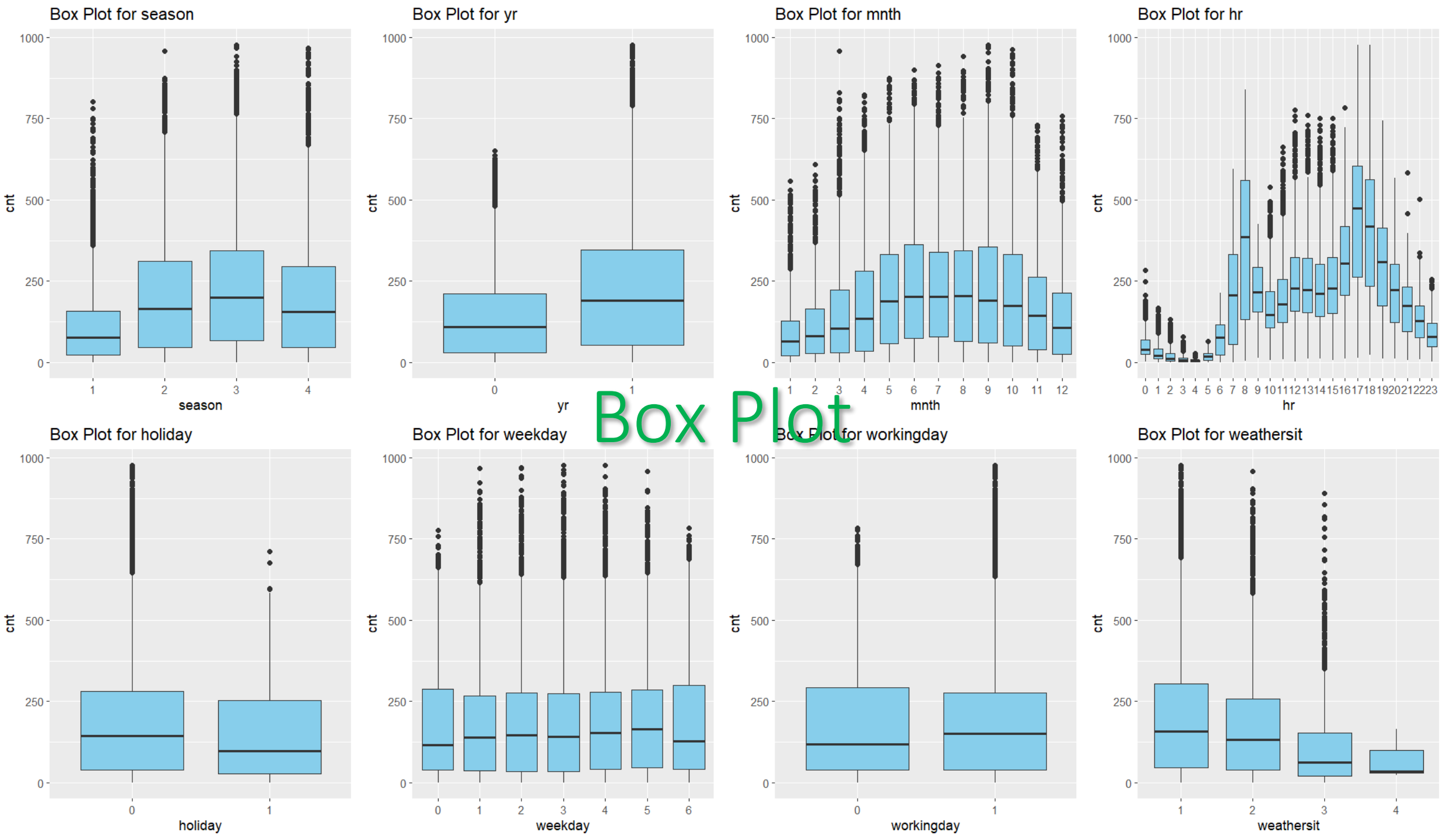
Bar Plot


Bar Plot for holiday
cnt
1000
750
500
250
0
0
1
holiday

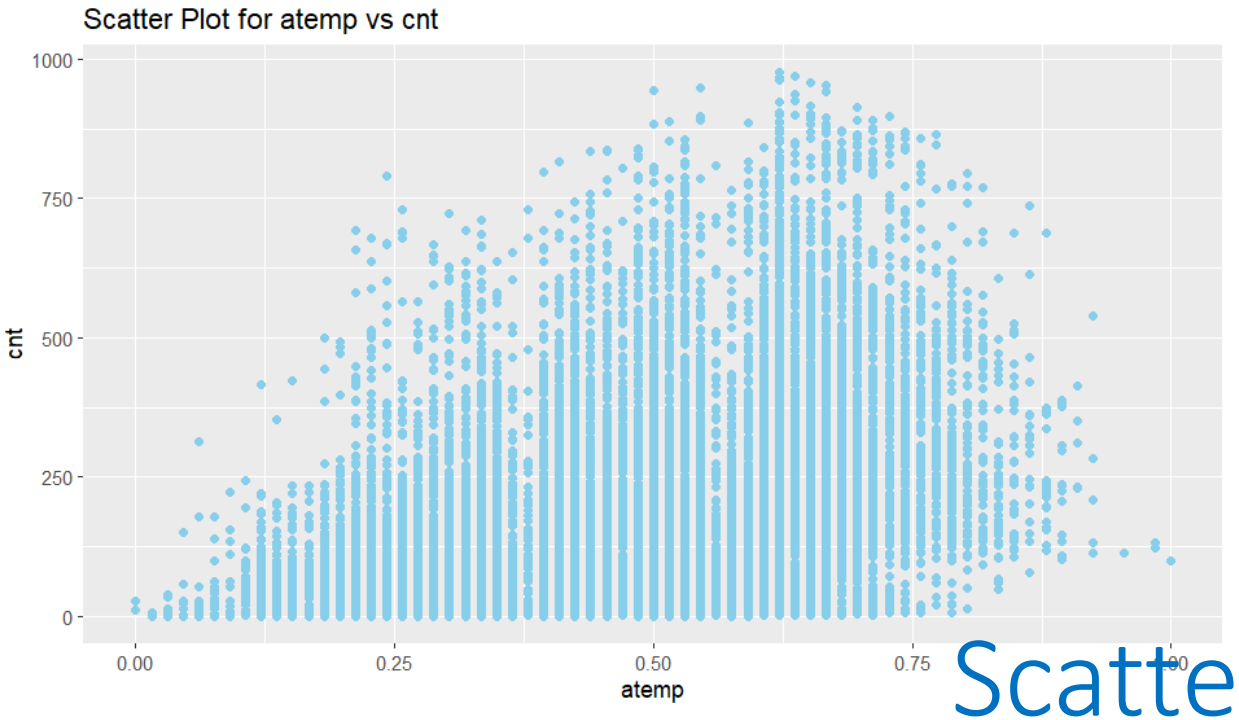

Bar Plot for weekday
cnt
1000
750
500
250
0
0 1 2 3 4 5 6
weekday

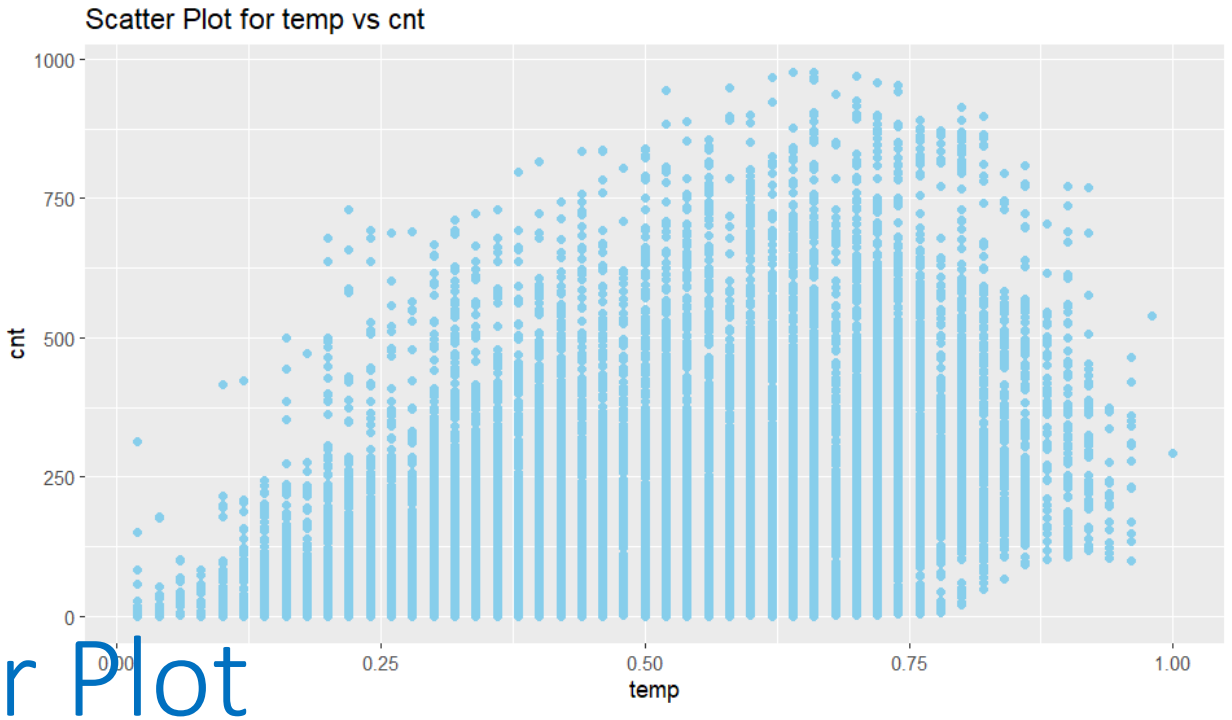

Bar Plot for workingday
cnt
1000
750
500
250
0
0
1
workingday

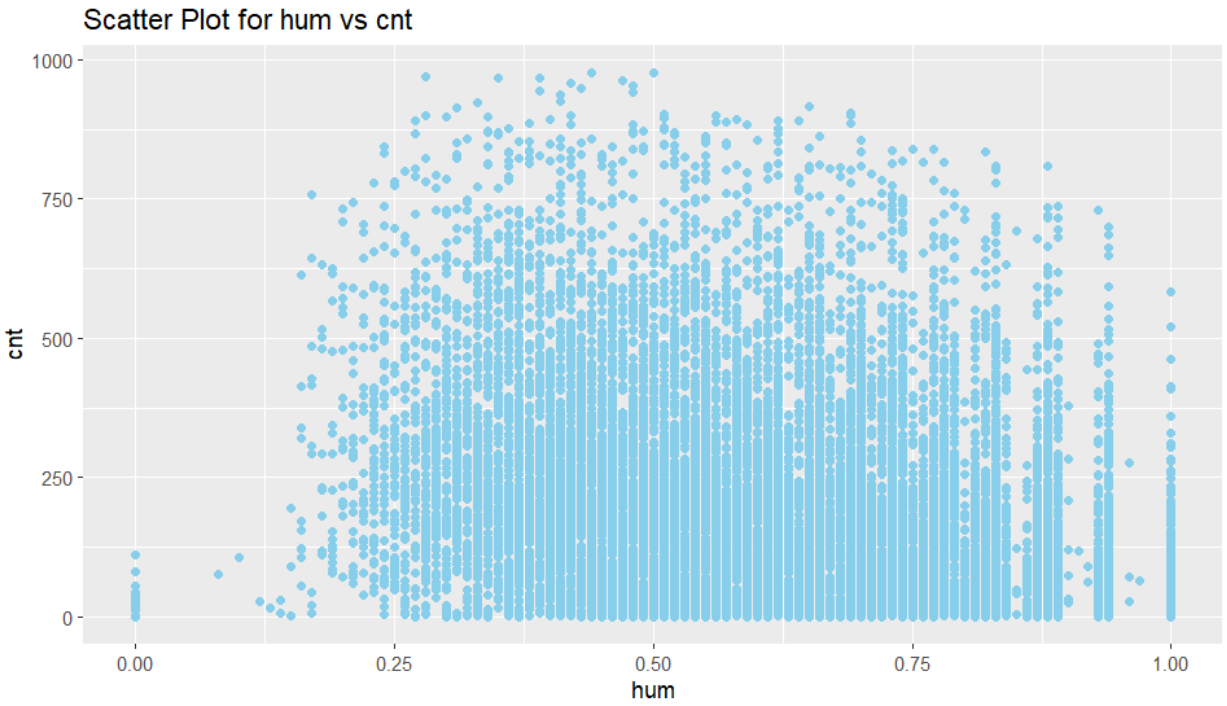

Bar Plot for weathersit
cnt
1000
750
500
250
0
1
2
3
4
weathersit

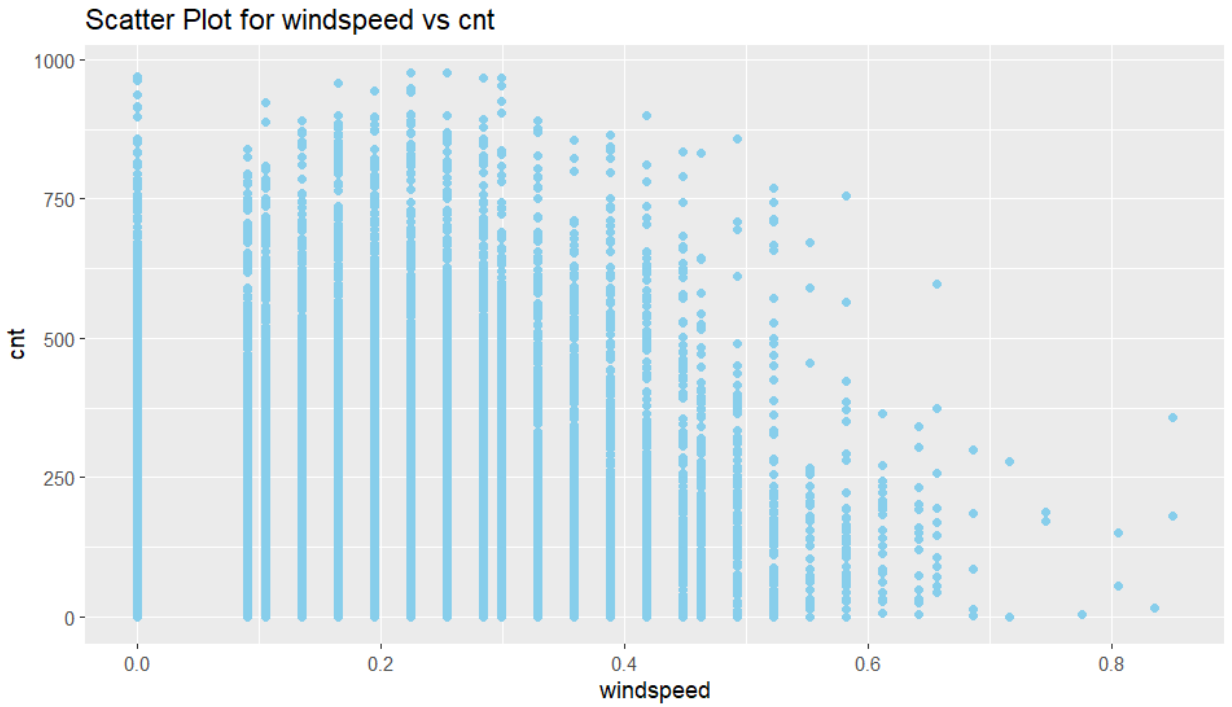

Box Plot for season
Box Plot for yr
Box Plot for mnth
Box Plot for hr
Box Plot
Box Plot for holiday
Box Plot for weekday
Box Plot for workingday
Box Plot for weathersit
cnt
season
yr
mnth
hr
holiday
weekday
workingday
weathersit

Scatter Plot

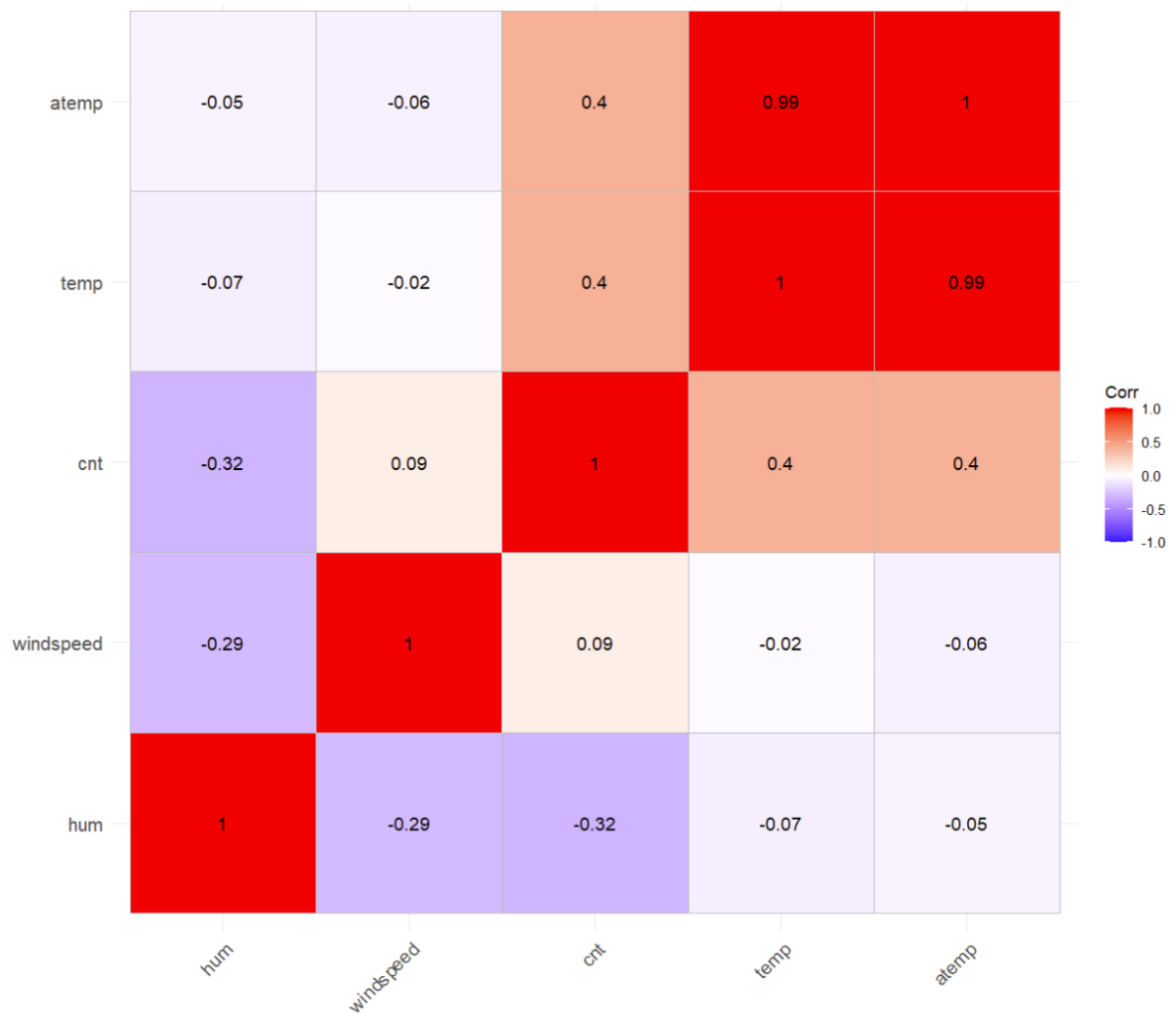




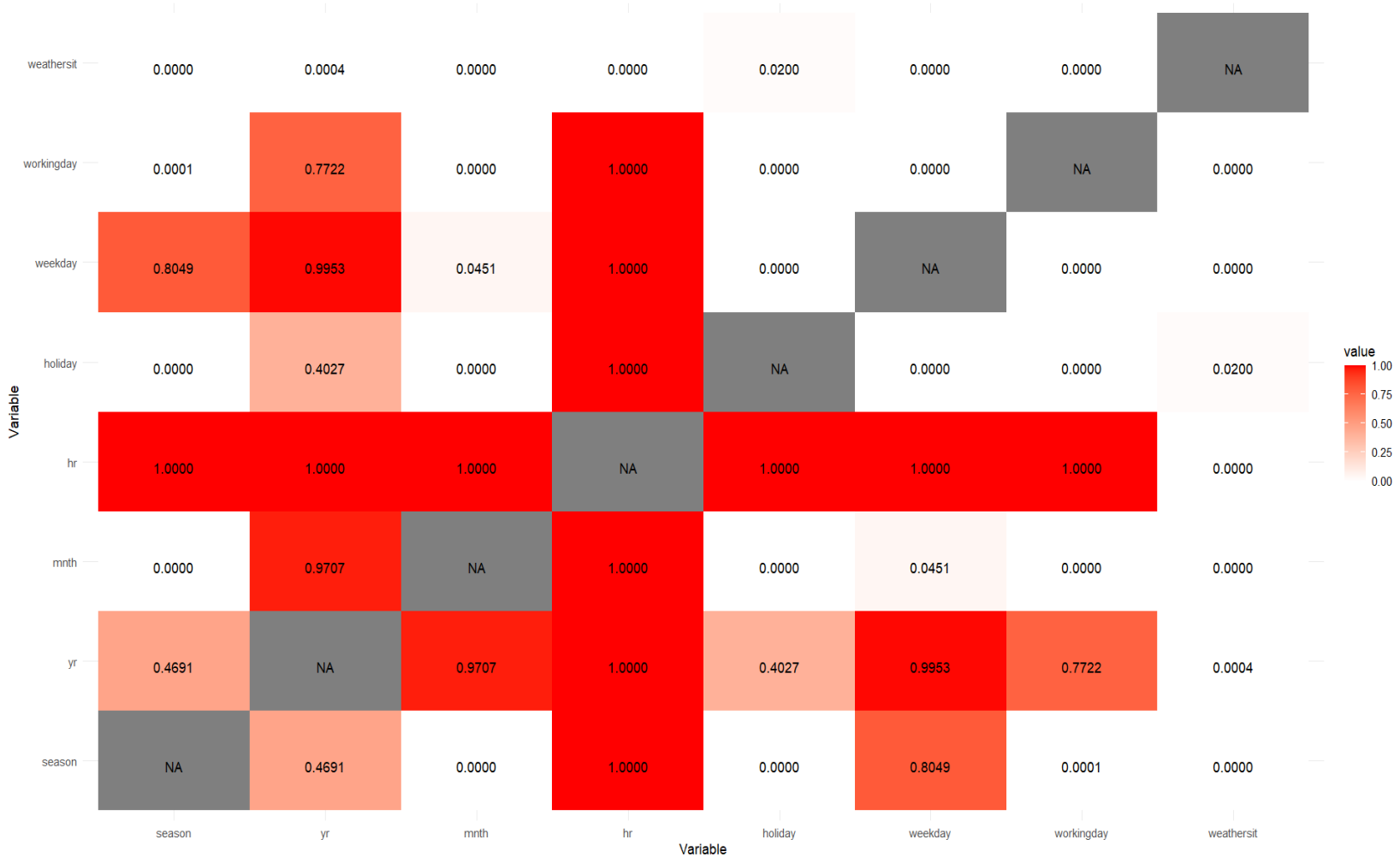




CORRELATION



CHI – SQUARE TEST



Transformation

```
> dim(dff)
[1] 17377    53
> colnames(dff)
 [1] "season2"    "season3"    "season4"    "yr1"        "mnth2"      "mnth3"      "mnth4"      "mnth5"
 [9] "mnth6"      "mnth7"      "mnth8"      "mnth9"      "mnth10"     "mnth11"     "mnth12"     "hr1"
[17] "hr2"        "hr3"        "hr4"        "hr5"        "hr6"        "hr7"        "hr8"        "hr9"
[25] "hr10"       "hr11"       "hr12"       "hr13"       "hr14"       "hr15"       "hr16"       "hr17"
[33] "hr18"       "hr19"       "hr20"       "hr21"       "hr22"       "hr23"       "holiday1"   "weekday1"
[41] "weekday2"   "weekday3"   "weekday4"   "weekday5"   "weekday6"   "weathersit2" "weathersit3" "weathersit4"
[49] "temp"       "atemp"      "hum"        "windspeed"  "cnt"
>
```

- Scaling using the Z normalization
  - ✓Improved convergence for gradient Descent
  - ✓Scale Invariance for some algorithmns
  - ✓Interpretability and comparisons
  - ✓Regularization Stability

- Identified and converted the categorical into factors and then converted into numerical using dummies.
- Scaled the numerical variable using the z Transformation.

ekday2	weekday3	weekday4	weekday5	weekday6	weathersit2	weathersit3	weathersit4	temp	atemp	hum	windspeed	cnt
-0.405313	-0.4075237	-0.4071396	-0.408675	2.4331070	-0.5950351	-0.2981875	-0.01314009	-1.3346564	-1.0932844	0.94738849	-1.55389398	16
-0.405313	-0.4075237	-0.4071396	-0.408675	2.4331070	-0.5950351	-0.2981875	-0.01314009	-1.4385237	-1.1817340	0.89555749	-1.55389398	40
-0.405313	-0.4075237	-0.4071396	-0.408675	2.4331070	-0.5950351	-0.2981875	-0.01314009	-1.4385237	-1.1817340	0.89555749	-1.55389398	32
-0.405313	-0.4075237	-0.4071396	-0.408675	2.4331070	-0.5950351	-0.2981875	-0.01314009	-1.3346564	-1.0932844	0.63640249	-1.55389398	13
-0.405313	-0.4075237	-0.4071396	-0.408675	2.4331070	-0.5950351	-0.2981875	-0.01314009	-1.3346564	-1.0932844	0.63640249	-1.55389398	1
-0.405313	-0.4075237	-0.4071396	-0.408675	2.4331070	1.6804765	-0.2981875	-0.01314009	-1.3346564	-1.2696017	0.63640249	-0.82152780	1
-0.405313	-0.4075237	-0.4071396	-0.408675	2.4331070	-0.5950351	-0.2981875	-0.01314009	-1.4385237	-1.1817340	0.89555749	-1.55389398	2
-0.405313	-0.4075237	-0.4071396	-0.408675	2.4331070	-0.5950351	-0.2981875	-0.01314009	-1.5423910	-1.2696017	1.20654350	-1.55389398	3
-0.405313	-0.4075237	-0.4071396	-0.408675	2.4331070	-0.5950351	-0.2981875	-0.01314009	-1.3346564	-1.0932844	0.63640249	-1.55389398	8
-0.405313	-0.4075237	-0.4071396	-0.408675	2.4331070	-0.5950351	-0.2981875	-0.01314009	-0.9191871	-0.7406499	0.68823349	-1.55389398	14
-0.405313	-0.4075237	-0.4071396	-0.408675	2.4331070	-0.5950351	-0.2981875	-0.01314009	-0.6075852	-0.4764649	0.68823349	0.51978124	36
-0.405313	-0.4075237	-0.4071396	-0.408675	2.4331070	-0.5950351	-0.2981875	-0.01314009	-0.7114525	-0.8290995	0.94738849	0.76417576	56
-0.405313	-0.4075237	-0.4071396	-0.408675	2.4331070	-0.5950351	-0.2981875	-0.01314009	-0.3998506	-0.3001476	0.74006449	0.76417576	84
-0.405313	-0.4075237	-0.4071396	-0.408675	2.4331070	1.6804765	-0.2981875	-0.01314009	-0.1921160	-0.1238303	0.48090948	0.88596433	94
-0.405313	-0.4075237	-0.4071396	-0.408675	2.4331070	1.6804765	-0.2981875	-0.01314009	-0.1921160	-0.1238303	0.48090948	0.76417576	106
-0.405313	-0.4075237	-0.4071396	-0.408675	2.4331070	1.6804765	-0.2981875	-0.01314009	-0.2959833	-0.2116980	0.74006449	0.88596433	110
-0.405313	-0.4075237	-0.4071396	-0.408675	2.4331070	1.6804765	-0.2981875	-0.01314009	-0.3998506	-0.3001476	0.99921949	0.88596433	93
-0.405313	-0.4075237	-0.4071396	-0.408675	2.4331070	1.6804765	-0.2981875	-0.01314009	-0.2959833	-0.2116980	0.99921949	0.76417576	67
-0.405313	-0.4075237	-0.4071396	-0.408675	2.4331070	-0.5950351	3.3534014	-0.01314009	-0.3998506	-0.3001476	1.31020550	0.51978124	35

Feature Selection

- “dteday” is a combination of “yr” and “mnth”.
- “instant” is index number
- “workingday” is associated with “holiday” and “weekday”
- “cnt” is also associated with “casual” and “registered”
- So, removed “dteday”, “instant”, “workingday”, “casual” and “registered.”

```
> colnames(data)
 [1] "instant"    "dteday"     "season"     "yr"         "mnth"       "hr"         "holiday"    "weekday"
 [9] "workingday" "weathersit" "temp"       "atemp"      "hum"        "windspeed"  "casual"     "registered"
[17] "cnt"
```

```
> colnames(cleaned_data)
 [1] "season"     "yr"         "mnth"       "hr"         "holiday"    "weekday"    "workingday" "weathersit"
 [9] "temp"       "atemp"      "hum"        "windspeed"  "cnt"
```

```
> colnames(dff)
 [1] "season2"    "season3"    "season4"    "yr1"        "mnth2"      "mnth3"      "mnth4"      "mnth5"
 [9] "mnth6"      "mnth7"      "mnth8"      "mnth9"      "mnth10"     "mnth11"     "mnth12"     "hr1"
[17] "hr2"        "hr3"        "hr4"        "hr5"        "hr6"        "hr7"        "hr8"        "hr9"
[25] "hr10"       "hr11"       "hr12"       "hr13"       "hr14"       "hr15"       "hr16"       "hr17"
[33] "hr18"       "hr19"       "hr20"       "hr21"       "hr22"       "hr23"       "holiday1"   "weekday1"
[41] "weekday2"   "weekday3"   "weekday4"   "weekday5"   "weekday6"   "weathersit2" "weathersit3" "weathersit4"
[49] "temp"       "atemp"      "hum"        "windspeed"  "cnt"
```


MODEL BUILDING
AND
EVALUATION

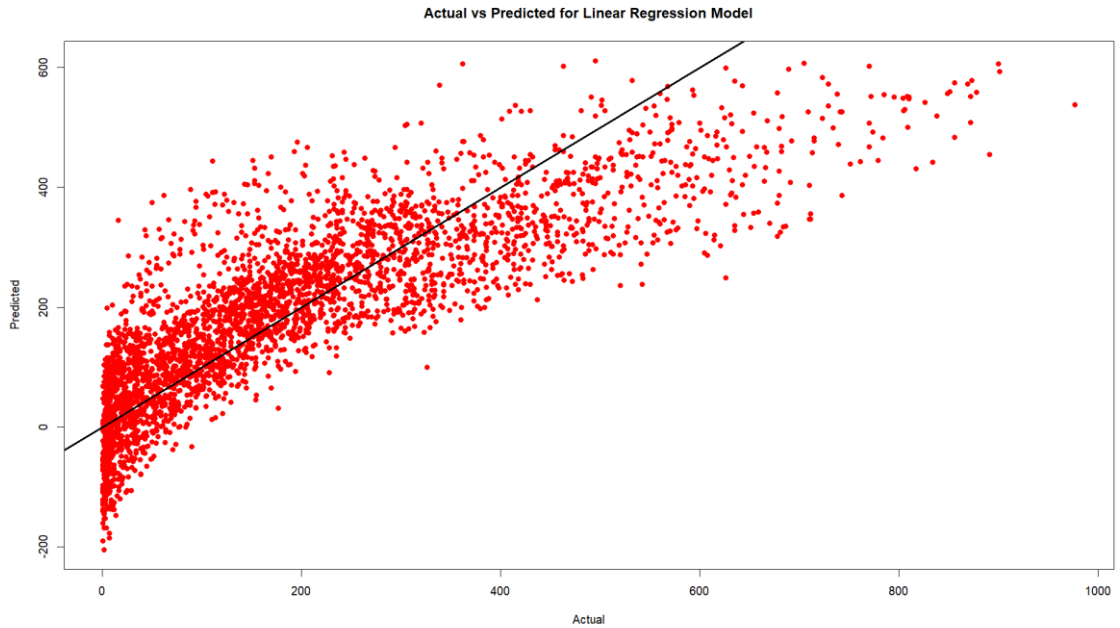
SIMPLE LINEAR REGRESSION

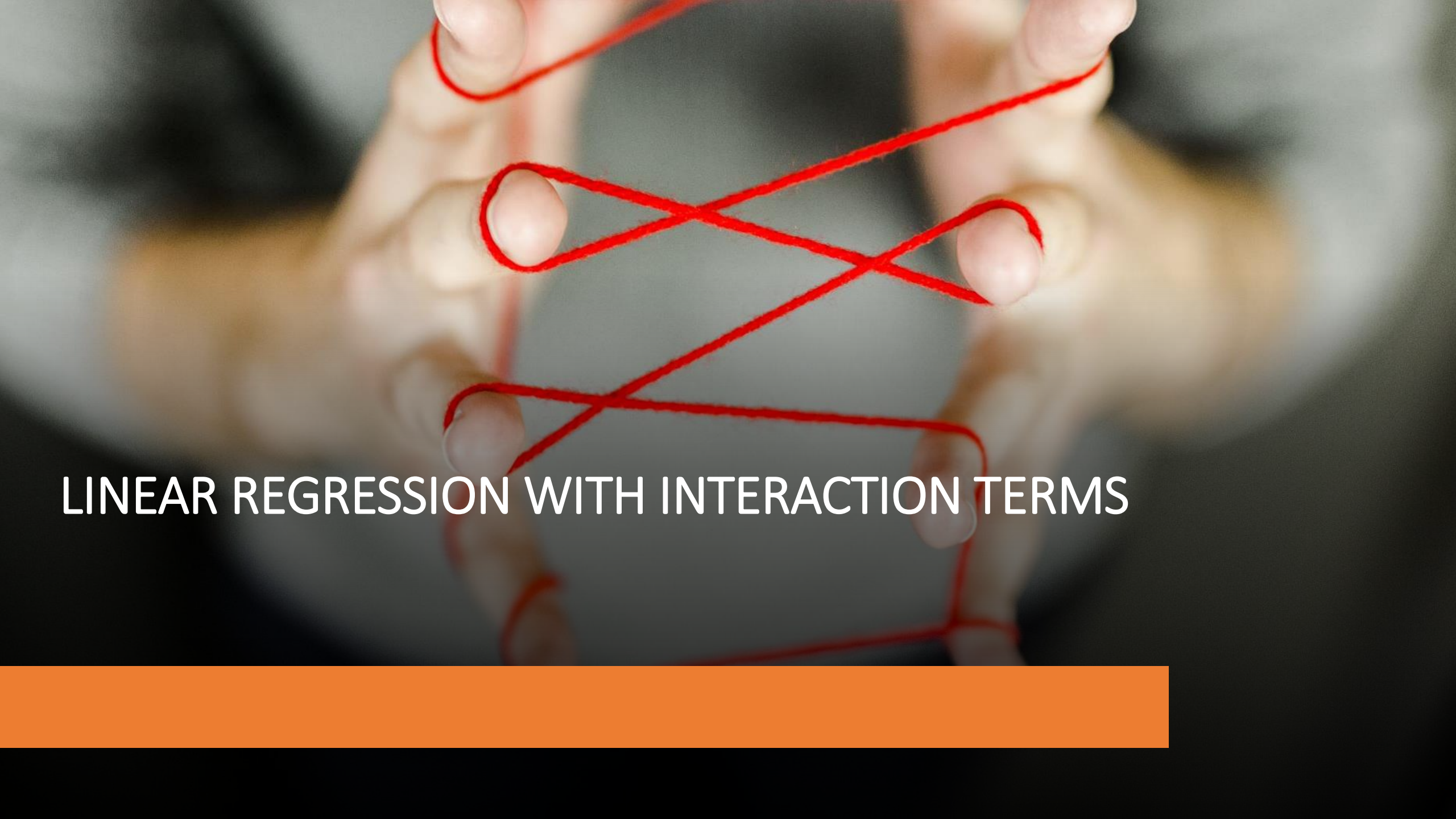
SIMPLE LINEAR REGRESSION

```
> lm.summary

Call:
lm(formula = .outcome ~ ., data = dat)

Residuals:
    Min      1Q  Median      3Q     Max
-398.75  -60.55   -7.77   51.05  434.65

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 189.3061     0.8689 217.856  < 2e-16 ***
season2      16.4860     2.3838   6.916 4.86e-12 ***
season3      13.5262     2.8407   4.762 1.94e-06 ***
season4      29.8907     2.3664  12.632  < 2e-16 ***
yr1          42.2000     0.8798  47.967  < 2e-16 ***
mnth2         1.3089     1.1856   1.104 0.269622
mnth3         4.5463     1.3910   3.268 0.001084 **
mnth4         2.3462     2.0319   1.155 0.248238
mnth5         5.9408     2.2091   2.689 0.007169 **
mnth6         2.0166     2.2367   0.902 0.367290
mnth7        -2.5806     2.5506  -1.012 0.311672
mnth8         3.0686     2.4793   1.238 0.215848
mnth9         9.5621     2.1750   4.396 1.11e-05 ***
weathersit2  -4.7862     0.9498  -5.039 4.73e-07 ***
weathersit3 -18.6500     1.0008 -18.635  < 2e-16 ***
weathersit4  -0.8511     0.7794  -1.092 0.274861
temp         25.5675     6.1963   4.126 3.71e-05 ***
atemp        18.0229     5.7242   3.149 0.001644 **
hum         -15.2839     1.2053 -12.681  < 2e-16 ***
windspeed    -2.6369     0.9762  -2.701 0.006918 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 102.4 on 13849 degrees of freedom
Multiple R-squared:  0.6844,    Adjusted R-squared:  0.6833
F-statistic: 577.6 on 52 and 13849 DF,  p-value: < 2.2e-16
```

```
> lm.fit
Linear Regression

13902 samples
   52 predictor

No pre-processing
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 11121, 11122, 11121, 11122, 11122
Resampling results:

  RMSE      Rsquared   MAE
  102.6258  0.6821557  75.82325
```

```
> cat("R-squared: ", r_squared_lm, "\n")
R-squared:  0.6929337
> cat("Mean Square Error (MsE): ", mse_lm, "\n")
Mean Square Error (MsE):  9841.131
> cat("Root Mean Square Error (RMSE): ", rmse_lm, "\n")
Root Mean Square Error (RMSE):  99.20247
```

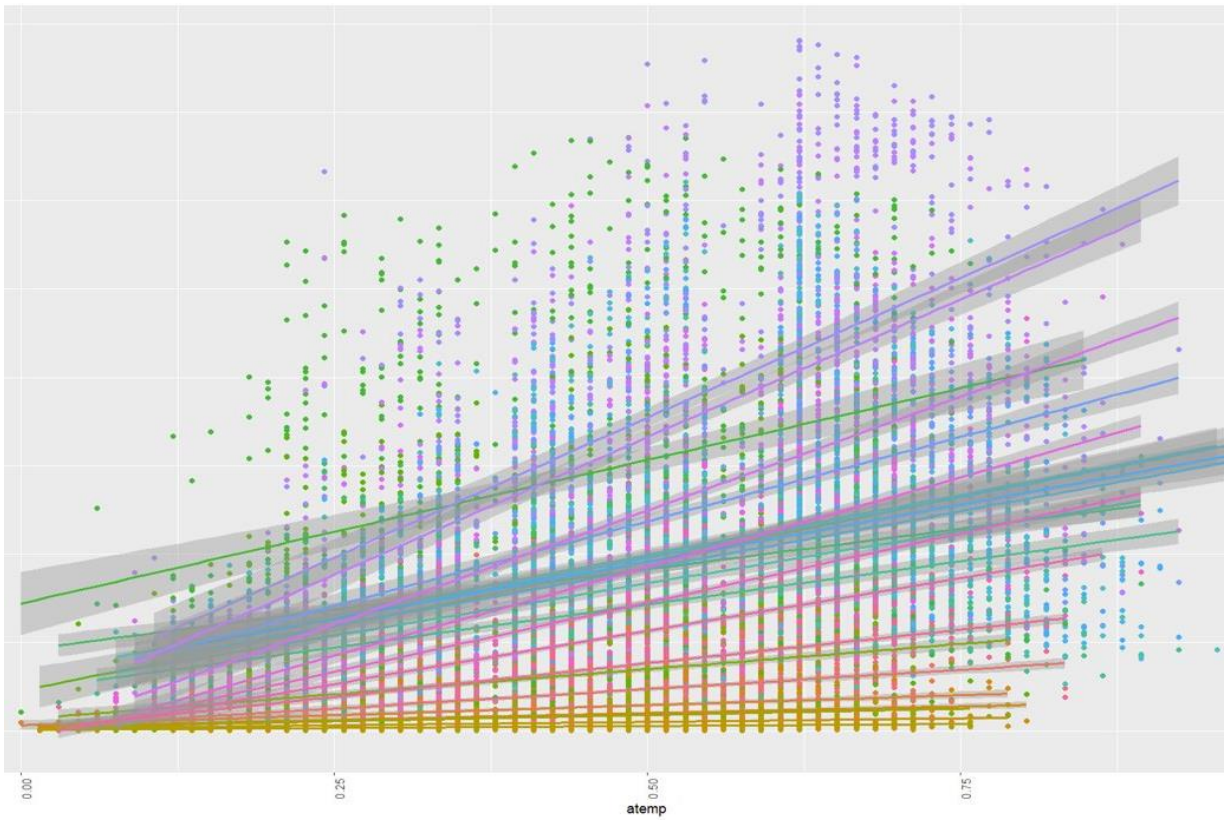




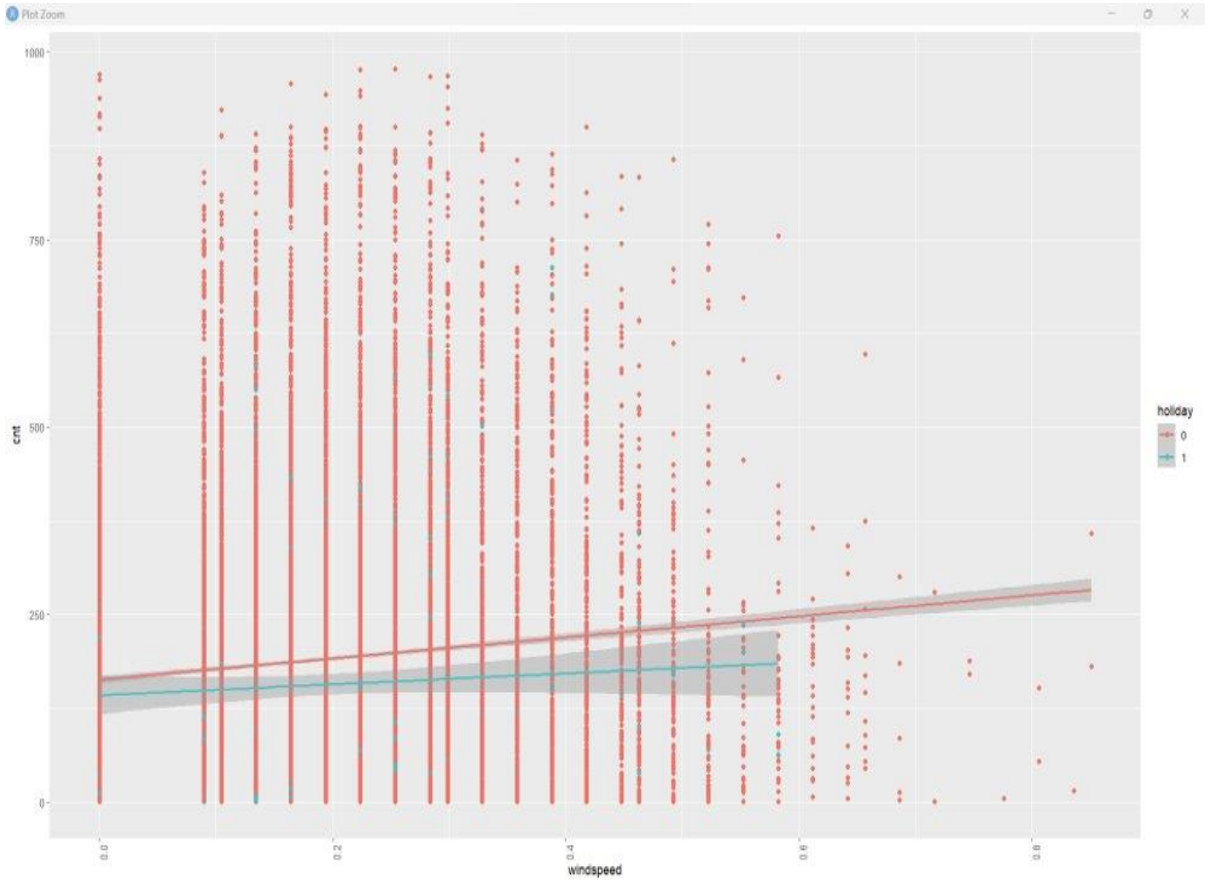
LINEAR REGRESSION WITH INTERACTION TERMS

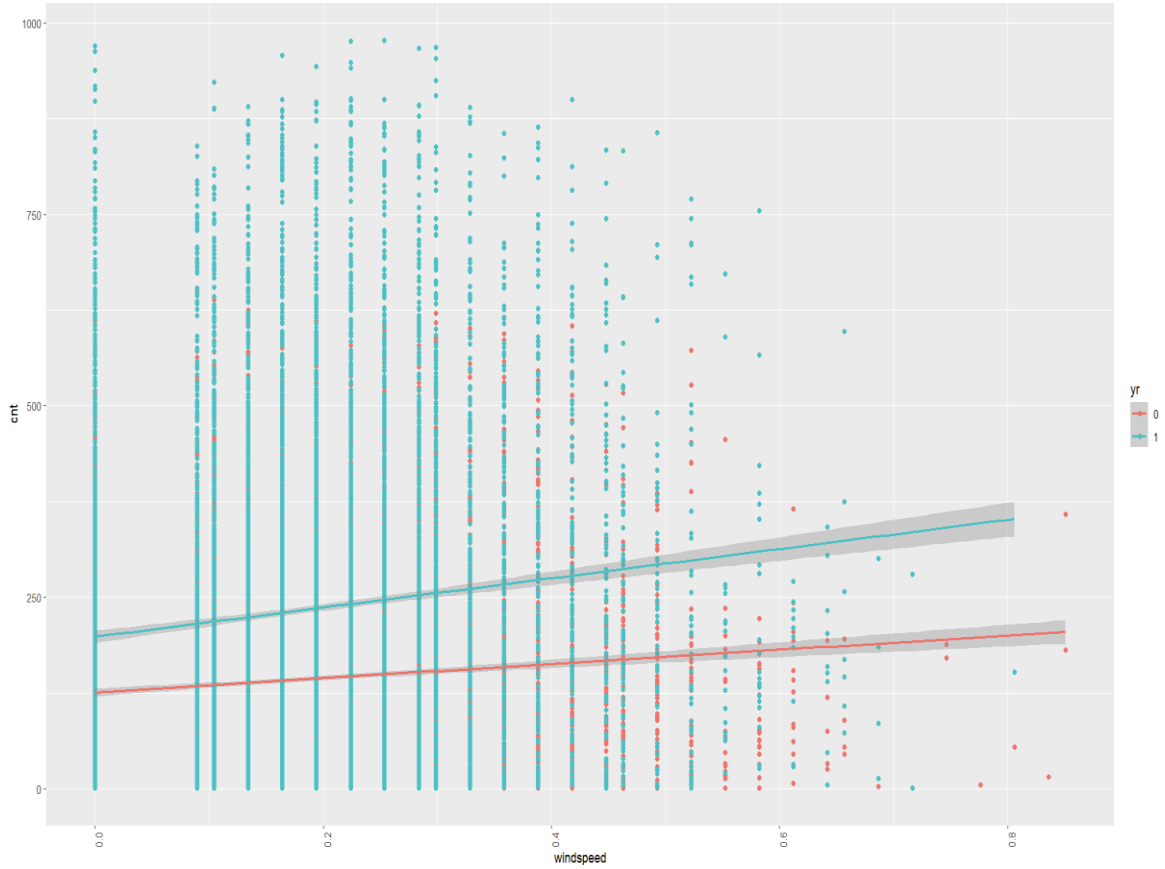
LINEAR REGRESSION WITH INTERACTION TERMS

INTERACTION TERM



NON - INTERACTION TERM





LINEAR REGRESSION WITH INTERACTION TERMS

- Following Interaction terms causing the Multicollinearity

- season*mnth
- season*weathersit
- mnth*holiday
- mnth*weathersit
- hr2*weathersit
- holiday*weekday
- holiday*weathersit
- weekday*weathersit
- weathersit*temp
- weathersit*atemp
- weathersit*hum
- weathersit*windspeed

- Finally considered the Interaction terms

- ✓ holiday*atemp
- ✓ temp*atemp
- ✓ season*yr
- ✓ yr*atemp
- ✓ hr*holiday
- ✓ season*hr
- ✓ yr*hr
- ✓ mnth*hr
- ✓ yr*hr
- ✓ mnth*hr
- ✓ yr*hr
- ✓ mnth*hum
- ✓ yr*mnth
- ✓ hr*weekday

LINEAR REGRESSION WITH INTERACTION TERMS

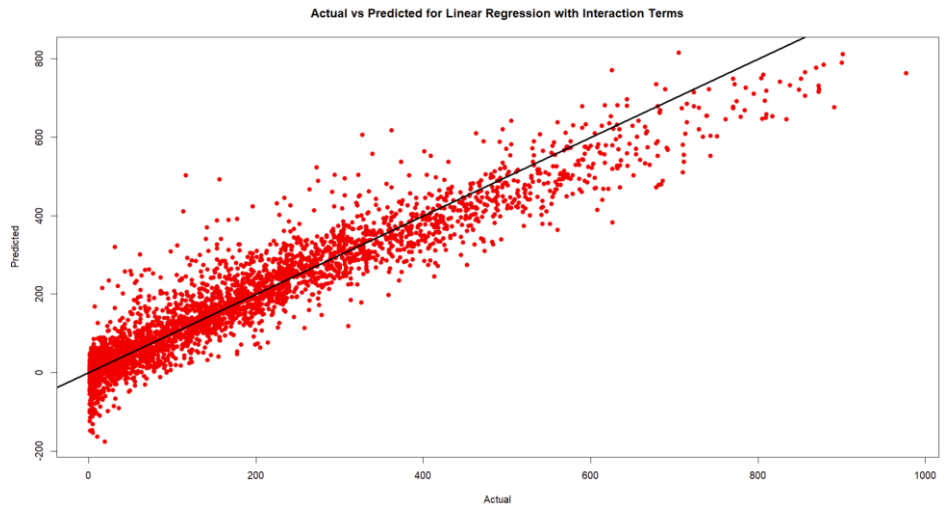
```
> lm.fit_it <- train(cnt ~.+(holiday*atemp)+(temp*atemp)+(season*yr)+(yr*atemp)+(hr*holiday)+(season*hr)+
+                      (yr*hr)+(mnth*hr)+(yr*hr)+(mnth*hr)+(yr*hr)+(mnth*hum)+(yr*mnth)+(hr*weekday),
+                    data = cleaned_data[trainIndex, ],method="lm",trControl = ctrl)
> lm.summary_it <- summary(lm.fit_it)
> lm.summary_it

Call:
lm(formula = .outcome ~ ., data = dat)

Residuals:
    Min      1Q  Median      3Q     Max
-471.46  -24.72   -0.66   27.37  458.02

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)        25.4113    12.4849   2.035 0.041835 *
season2            -6.8703    14.7611  -0.465 0.641627
season3            -5.4973    18.4210  -0.298 0.765385
season4            -1.6012    16.1243  -0.099 0.920897
yr1               -47.1773     6.6346  -7.111 1.21e-12 ***
mnth2             -25.7637    14.8869  -1.731 0.083542 .
mnth3             -43.4961    15.4770  -2.810 0.004956 **
mnth4              12.1285    21.4464   0.566 0.571724
mnth5              66.0083    22.5466   2.928 0.003421 **
mnth6              21.5619    21.8351   0.987 0.323421
mnth7             -28.5266    25.2783  -1.129 0.259129
mnth8              47.7246    25.3290   1.884 0.059562 .
mnth9             141.9655    24.9531   5.689 1.30e-08 ***
mnth10             96.3250    23.4073   4.115 3.89e-05 ***

`mnth11:hr2`      -11.5817    27.6573  -0.419 0.675400
`mnth12:hr2`      -13.1250    22.3477  -0.587 0.557004
`mnth2:hr3`        -4.2622    17.7005  -0.241 0.809718
`mnth3:hr3`        -7.6630    18.4342  -0.416 0.677642
`mnth4:hr3`        -0.5897    26.7212  -0.022 0.982392
`mnth5:hr3`       -25.5080    26.5778  -0.960 0.337200
 [ reached getOption("max.print") -- omitted 387 rows ]
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 57.79 on 13315 degrees of freedom
Multiple R-squared:  0.9034,    Adjusted R-squared:  0.8991
F-statistic: 212.5 on 586 and 13315 DF,  p-value: < 2.2e-16
```

MODEL

```
> lm.fit_it
Linear Regression

13902 samples
   11 predictor

No pre-processing
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 11120, 11122, 11122, 11122, 11122
Resampling results:

  RMSE      Rsquared   MAE
  59.62651  0.8926881  40.74902
```

METRICS WITH TEST DATA

```
> cat("R-squared: ", r_squared_it, "\n")
R-squared:  0.8994243
> cat("Mean Square Error (MsE): ", mse_it, "\n")
Mean Square Error (MsE):  3223.339
> cat("Root Mean Square Error (RMSE): ", rmse_it, "\n")
Root Mean Square Error (RMSE):  56.77445
>
```



LINEAR REGRESSION WITH POLYNOMIAL DEGREE

LINEAR REGRESSION WITH POLYNOMIAL DEGREE

Results for degree 2

```
> lm.fit_2
Linear Regression

13902 samples
   56 predictor

No pre-processing
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 11121, 11122, 11121, 11122, 11122
Resampling results:

  RMSE      Rsquared   MAE
  102.4031  0.6834022  75.65933
```

Results for degree 4

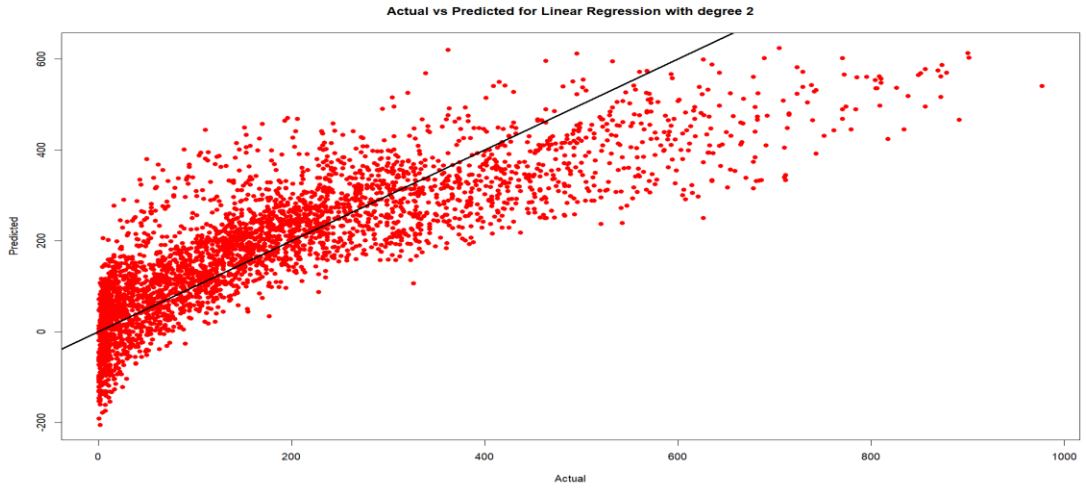
```
> lm.fit_4
Linear Regression

13902 samples
   64 predictor

No pre-processing
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 11122, 11121, 11122, 11122, 11121
Resampling results:

  RMSE      Rsquared   MAE
  101.0793  0.6915714  75.20426
```

Results for degree 3

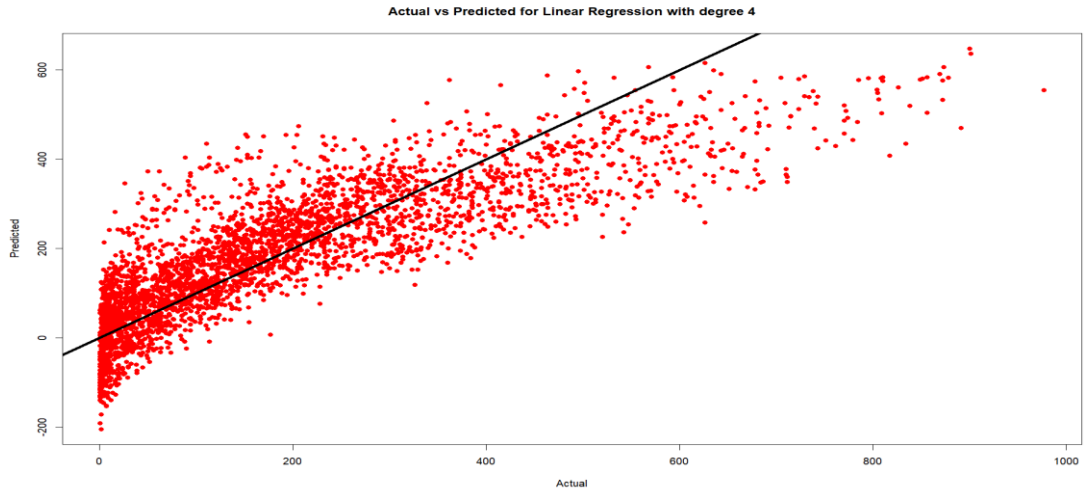
```
> lm.fit_3
Linear Regression

13902 samples
   60 predictor

No pre-processing
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 11122, 11121, 11121, 11122, 11122
Resampling results:

  RMSE      Rsquared   MAE
  101.2993  0.6901062  75.29893
```

LINEAR REGRESSION WITH POLYNOMIAL DEGREE

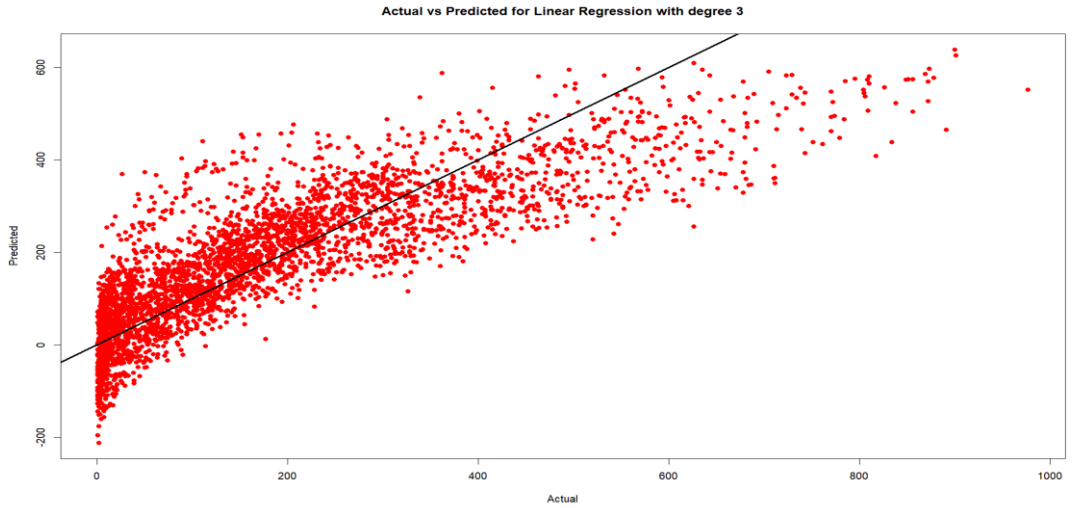
Results for degree 2



Results for degree 4



Results for degree 3



LINEAR REGRESSION WITH POLYNOMIAL DEGREE

Results for degree 2

```
> cat("R-squared: ", r_squared_lm2, "\n")
R-squared:  0.6954175
> cat("Mean Square Error (MSE): ", mse_lm2, "\n")
Mean Square Error (MSE):  9761.529
> cat("Root Mean Square Error (RMSE): ", rmse_lm2, "\n")
Root Mean Square Error (RMSE):  98.80045
```

Results for degree 4

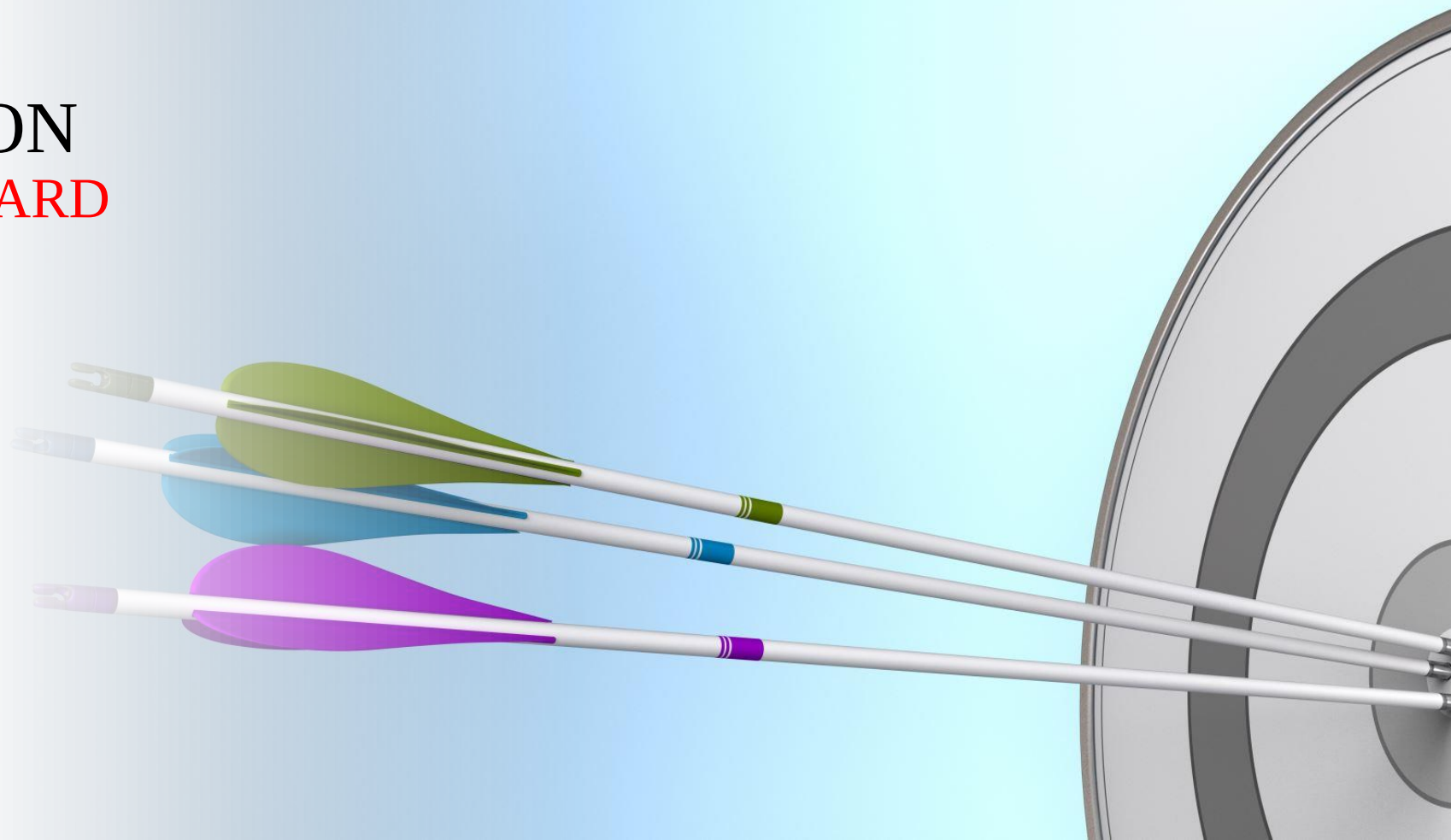
```
> cat("R-squared: ", r_squared_lm4, "\n")
R-squared:  0.7031851
> cat("Mean Square Error (MsE): ", mse_lm4, "\n")
Mean Square Error (MsE):  9512.586
> cat("Root Mean Square Error (RMSE): ", rmse_lm4, "\n")
Root Mean Square Error (RMSE):  97.53249
```

Results for degree 3

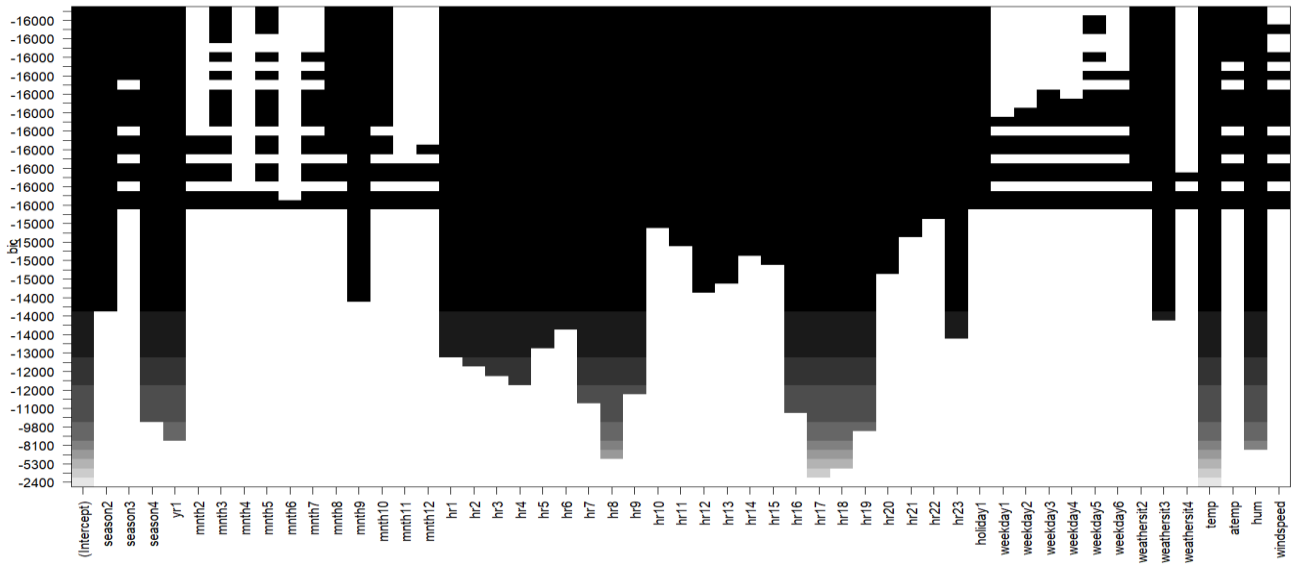
```
> cat("R-squared: ", r_squared_lm3, "\n")
R-squared:  0.7011467
> cat("Mean Square Error (MsE): ", mse_lm3, "\n")
Mean Square Error (MsE):  9577.911
> cat("Root Mean Square Error (RMSE): ", rmse_lm3, "\n")
Root Mean Square Error (RMSE):  97.8668
```

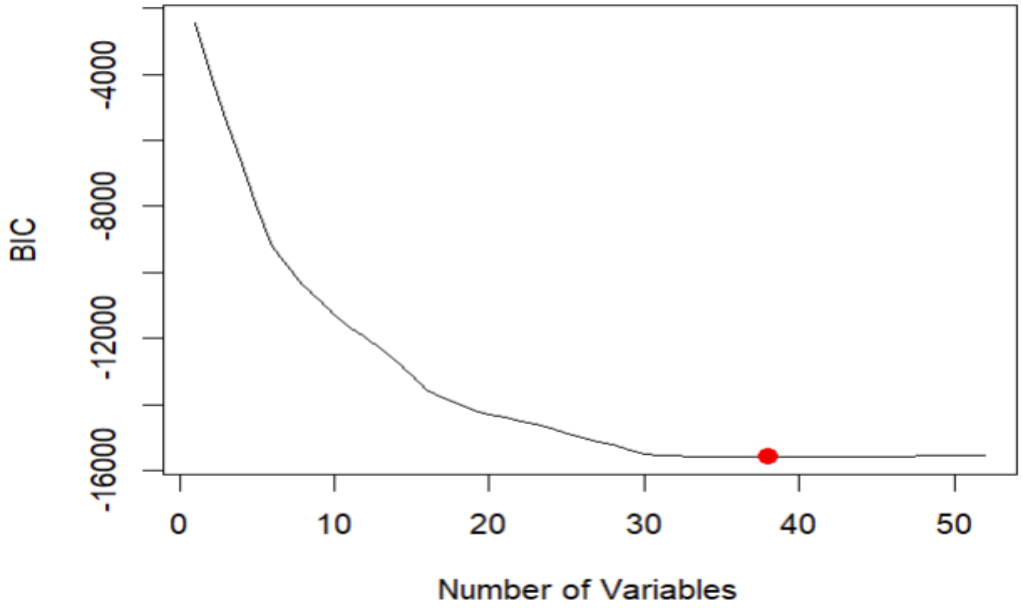
SUBSET SELECTION

FORWARD AND BACKWARD SELECTION



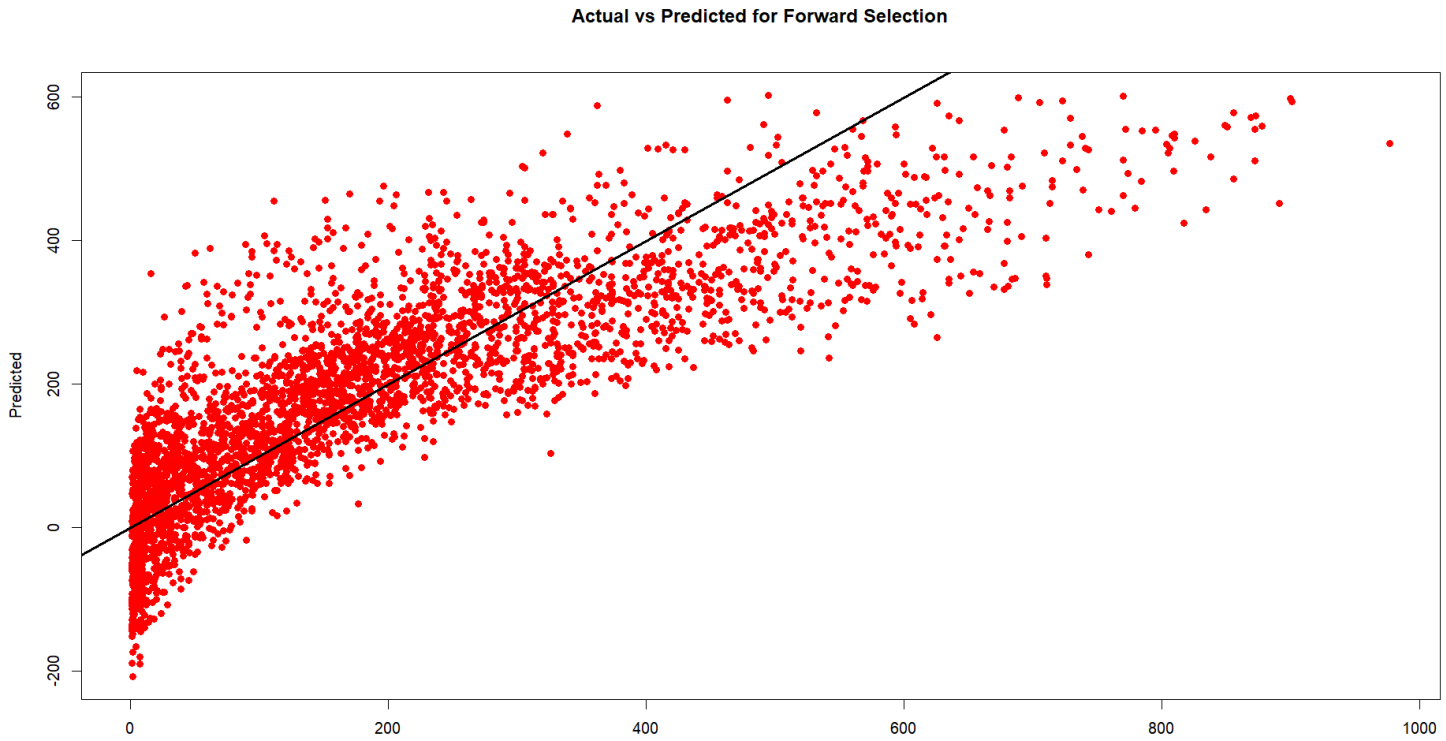
SUBSET SELECTION
FORWARD SELECTION






```
> coef(regfit.fwd,38)
(Intercept)     season2     season3     season4         yr1       mnth3       mnth5       mnth8       mnth9
 189.288952   18.747791   10.223475   25.909799   42.103023    3.584787    3.924685    4.702713   11.367324
     mnth10         hr1         hr2         hr3         hr4         hr5         hr6         hr7         hr8
   5.544430   -3.209394   -5.243933   -6.922451   -8.122287   -4.202379    7.272126   34.115324   62.272467
        hr9        hr10        hr11        hr12        hr13        hr14        hr15        hr16        hr17
  32.447657   21.261054   26.891246   34.650584   33.425537   30.280654   32.611738   44.998609   76.803306
       hr18        hr19        hr20        hr21        hr22        hr23    holiday1  weathersit2  weathersit3
  68.946476   46.829041   31.178625   21.679811   14.347000    6.588060   -4.361285   -4.638782  -18.646796
       temp       atemp         hum
  24.600633   19.841455  -15.254342
```

FORWARD SELECTION

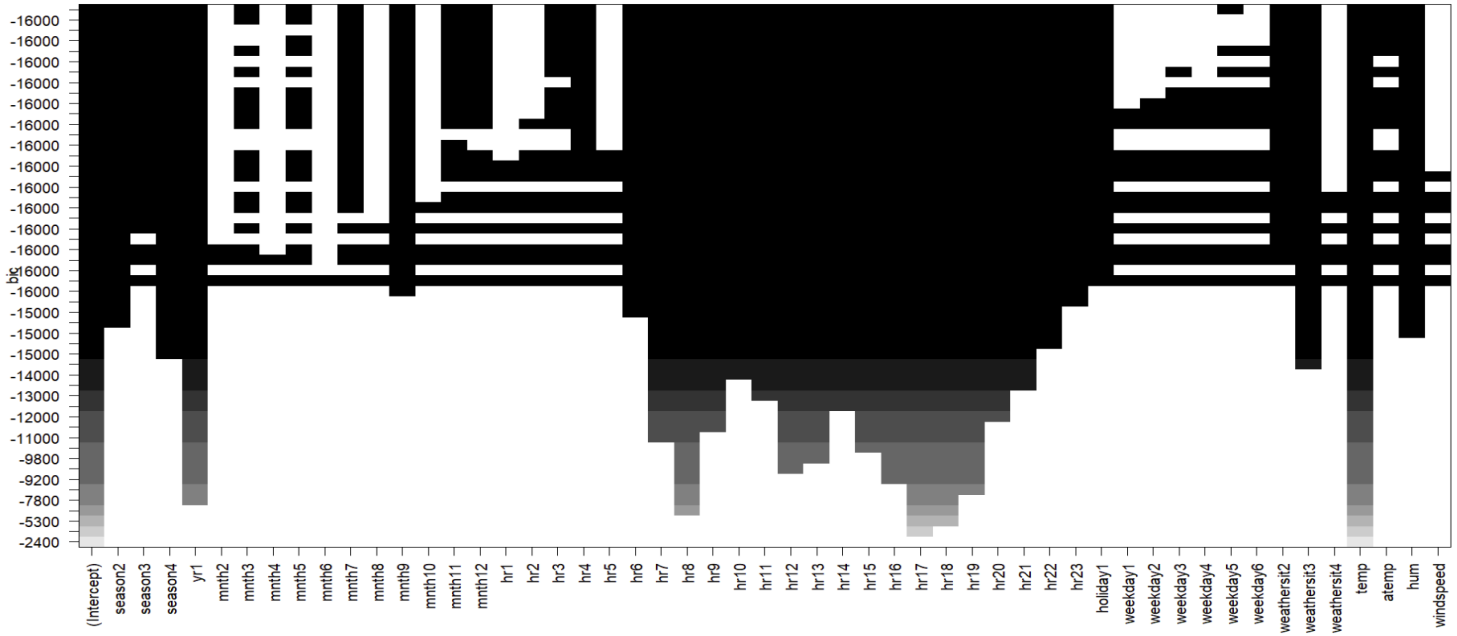


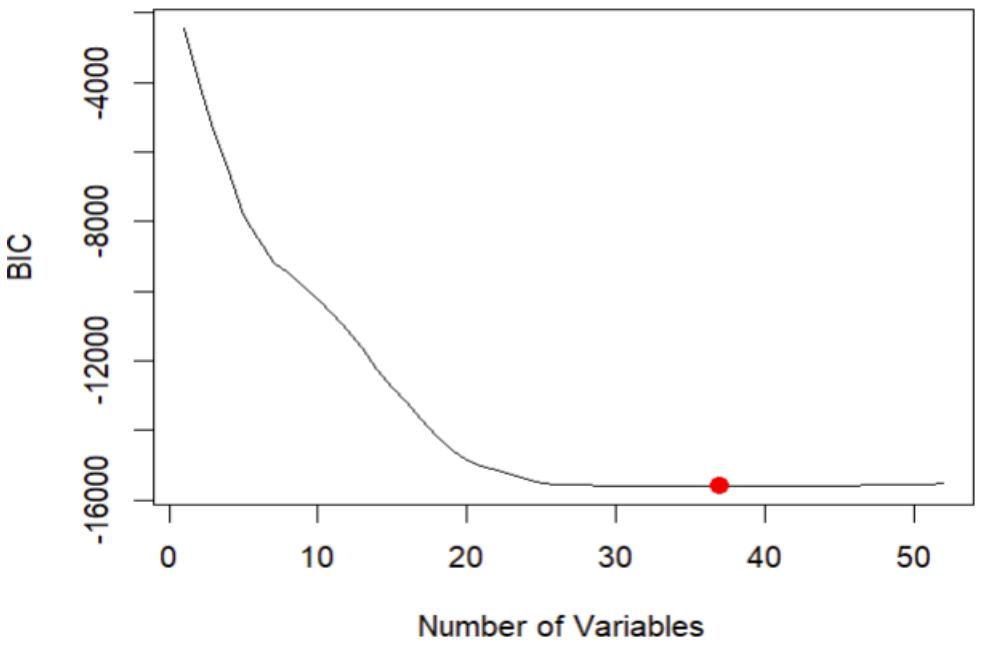

```
> cat("R-squared: ", r_squared_fw, "\n")
R-squared:  0.6909245
> cat("Mean Square Error (MsE): ", mse_fw, "\n")
Mean Square Error (MsE):  9905.524
> cat("Root Mean Square Error (RMSE): ", rmse_fw, "\n")
Root Mean Square Error (RMSE):  99.5265
```

```
> cv.errors.fwd
 [1] 26444.648 24318.891 22022.468 20428.590 18292.142 16597.379 15734.361 15044.118 14638.365 14045.652 13643.904 13355.286 12989.082 12584.272
[15] 12214.863 11711.843 11519.840 11342.025 11218.695 11078.790 11015.072 10923.049 10823.611 10693.821 10637.727 10504.583 10406.582 10334.992
[29] 10174.868 10083.815 10045.140 10028.980 10016.478  9966.989  9956.881  9923.796  9925.677  9905.524  9901.974  9875.581  9858.089  9846.435
[43]  9848.432  9837.912  9837.510  9834.092  9837.029  9842.000  9839.138  9839.497  9840.846
> rmse_fwd
 [1] 162.61811 155.94515 148.39969 142.92862 135.24845 128.83082 125.43668 122.65447 120.98911 118.51435 116.80712 115.56507 113.96965 112.17964
[15] 110.52087 108.22127 107.33052 106.49894 105.91834 105.25583 104.95271 104.51339 104.03658 103.41093 103.13936 102.49187 102.01265 101.66116
[29] 100.87055 100.41820 100.22545 100.14479 100.08235  99.83481  99.78417  99.61825  99.62769  99.52650  99.50866  99.37596  99.28791  99.22920
[43]  99.23927  99.18625  99.18422  99.16699  99.18180  99.20685  99.19243  99.19424  99.20104
> cv.r2.fwd
 [1] 0.1748651 0.2411937 0.3128474 0.3625802 0.4292423 0.4821229 0.5090511 0.5305883 0.5432487 0.5617428 0.5742782 0.5832838 0.5947102 0.6073413
[15] 0.6188677 0.6345631 0.6405540 0.6461023 0.6499504 0.6543158 0.6563040 0.6591753 0.6622780 0.6663277 0.6680780 0.6722324 0.6752903 0.6775241
[29] 0.6825203 0.6853614 0.6865681 0.6870724 0.6874624 0.6890066 0.6893220 0.6903543 0.6902956 0.6909245 0.6910352 0.6918587 0.6924045 0.6927682
[43] 0.6927059 0.6930341 0.6930467 0.6931533 0.6930617 0.6929066 0.6929959 0.6929847 0.6929426
```

SUBSET SELECTION

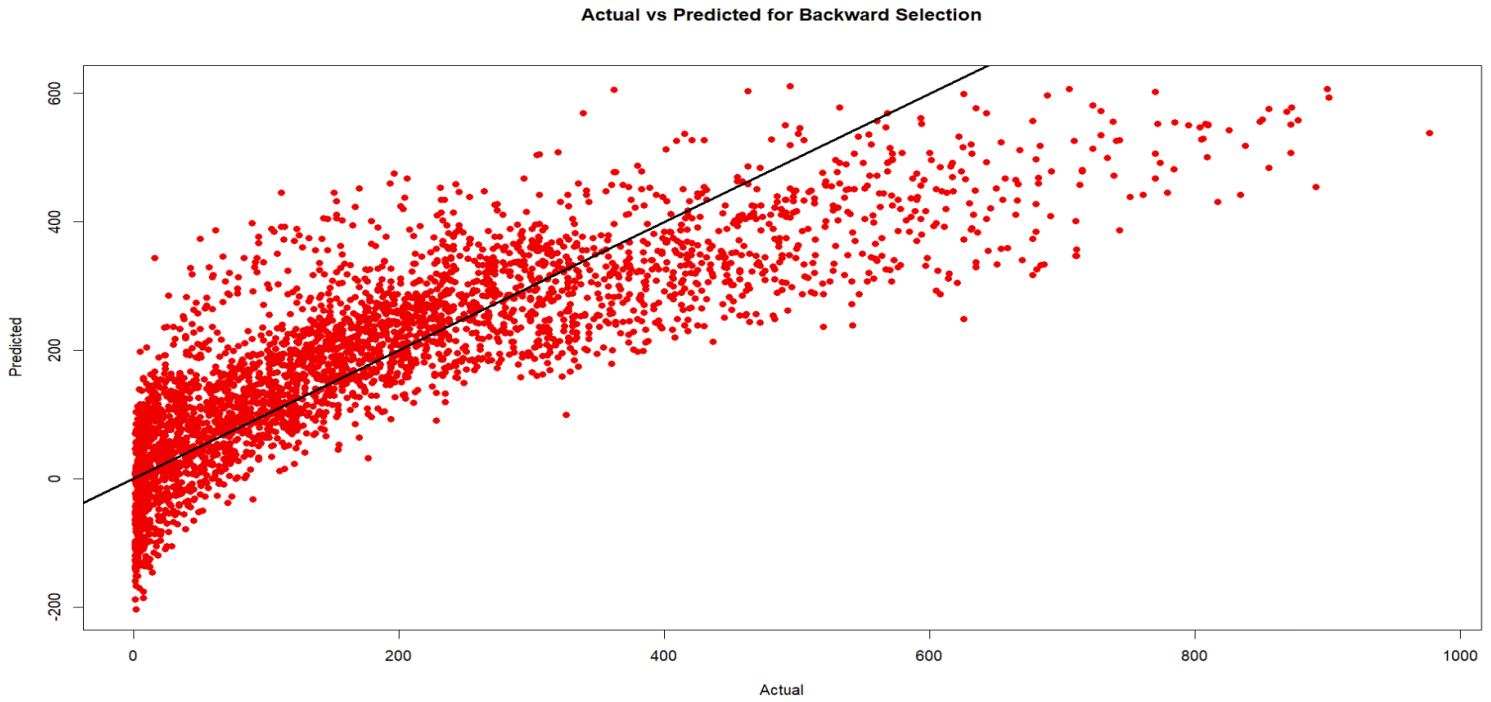
BACKWARD SELECTION






```
> coef(regfit.fwd,38)
(Intercept)     season2     season3     season4         yr1       mnth3       mnth5       mnth8       mnth9
 189.288952   18.747791   10.223475   25.909799   42.103023    3.584787    3.924685    4.702713   11.367324
     mnth10         hr1         hr2         hr3         hr4         hr5         hr6         hr7         hr8
   5.544430   -3.209394   -5.243933   -6.922451   -8.122287   -4.202379    7.272126   34.115324   62.272467
         hr9        hr10        hr11        hr12        hr13        hr14        hr15        hr16        hr17
  32.447657   21.261054   26.891246   34.650584   33.425537   30.280654   32.611738   44.998609   76.803306
        hr18        hr19        hr20        hr21        hr22        hr23    holiday1 weathersit2 weathersit3
  68.946476   46.829041   31.178625   21.679811   14.347000    6.588060   -4.361285   -4.638782  -18.646796
        temp       atemp         hum
  24.600633   19.841455  -15.254342
```

BACKWARD SELECTION




```
> cat("R-squared: ", r_squared_bw, "\n")
R-squared:  0.6906202
> cat("Mean Square Error (MsE): ", mse_bw, "\n")
Mean Square Error (MsE):  9915.277
> cat("Root Mean Square Error (RMSE): ", rmse_bw, "\n")
Root Mean Square Error (RMSE):  99.57548
```

```
> cv.errors.bwd
 [1] 26444.648 24318.891 22022.468 20428.590 18488.851 17397.522 16699.877 16275.440 15902.588 15602.269 15092.865 14563.414 13947.860 13212.865
[15] 12677.919 12256.154 11730.837 11316.816 10918.783 10667.190 10517.273 10385.000 10290.135 10184.860 10119.078 10081.399 10064.493 10056.063
[29] 10018.403 10018.208  9991.928  9986.486  9970.173  9934.566  9918.827  9918.969  9915.277  9903.889  9905.291  9893.876  9892.542  9890.012
[43]  9888.841  9878.748  9869.655  9844.325  9844.702  9836.636  9838.494  9839.497  9840.846
> rmse_bwd
 [1] 162.61811 155.94515 148.39969 142.92862 135.97372 131.89967 129.22801 127.57523 126.10547 124.90904 122.85302 120.67897 118.10106 114.94723
[15] 112.59626 110.70751 108.30899 106.38053 104.49298 103.28209 102.55376 101.90682 101.44031 100.92007 100.59363 100.40617 100.32195 100.27992
[29] 100.09197 100.09100  99.95963  99.93241  99.85075  99.67229  99.59331  99.59402  99.57548  99.51828  99.52533  99.46797  99.46126  99.44854
[43]  99.44265  99.39189  99.34614  99.21857  99.22047  99.17982  99.18918  99.19424  99.20104
> cv.r2.bwd
 [1] 0.1748651 0.2411937 0.3128474 0.3625802 0.4231046 0.4571566 0.4789247 0.4921682 0.5038020 0.5131727 0.5290673 0.5455874 0.5647941 0.5877277
[15] 0.6044193 0.6175793 0.6339704 0.6468888 0.6593084 0.6671587 0.6718365 0.6759637 0.6789237 0.6822085 0.6842611 0.6854368 0.6859643 0.6862273
[29] 0.6874024 0.6874084 0.6882285 0.6883982 0.6889073 0.6900183 0.6905094 0.6905049 0.6906202 0.6909755 0.6909317 0.6912879 0.6913295 0.6914085
[43] 0.6914450 0.6917599 0.6920436 0.6928340 0.6928222 0.6930739 0.6930160 0.6929847 0.6929426
```

REGULARIZATION
OR
SHRINKAGE TECHNIQUES
LASSO AND RIDGE REGRESSION

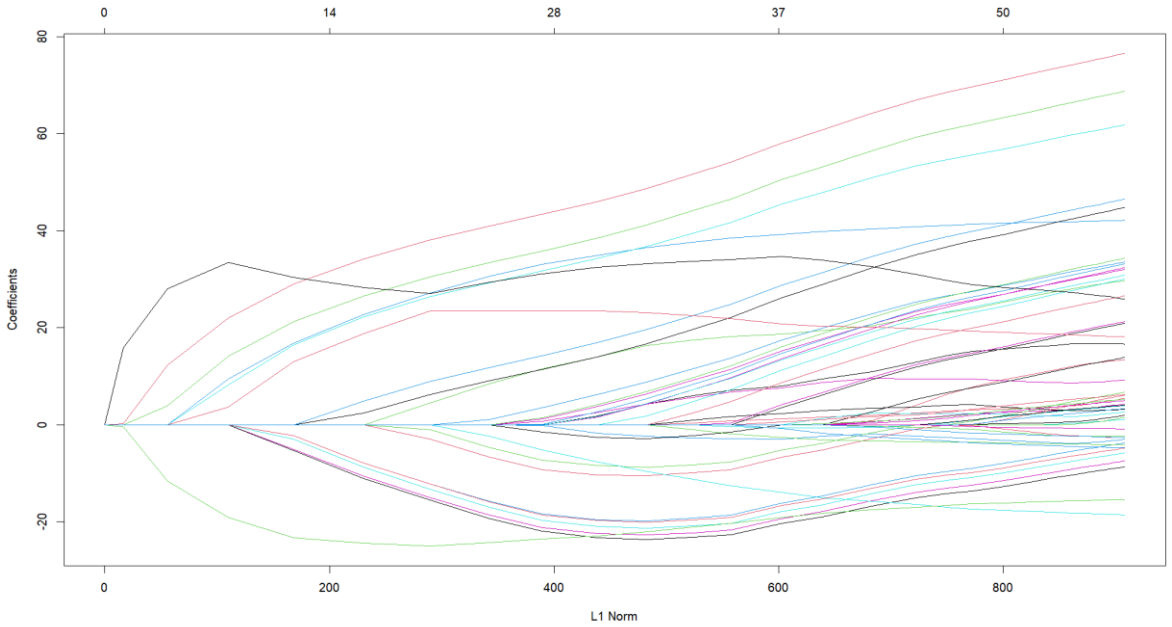
LASSO REGRESSION

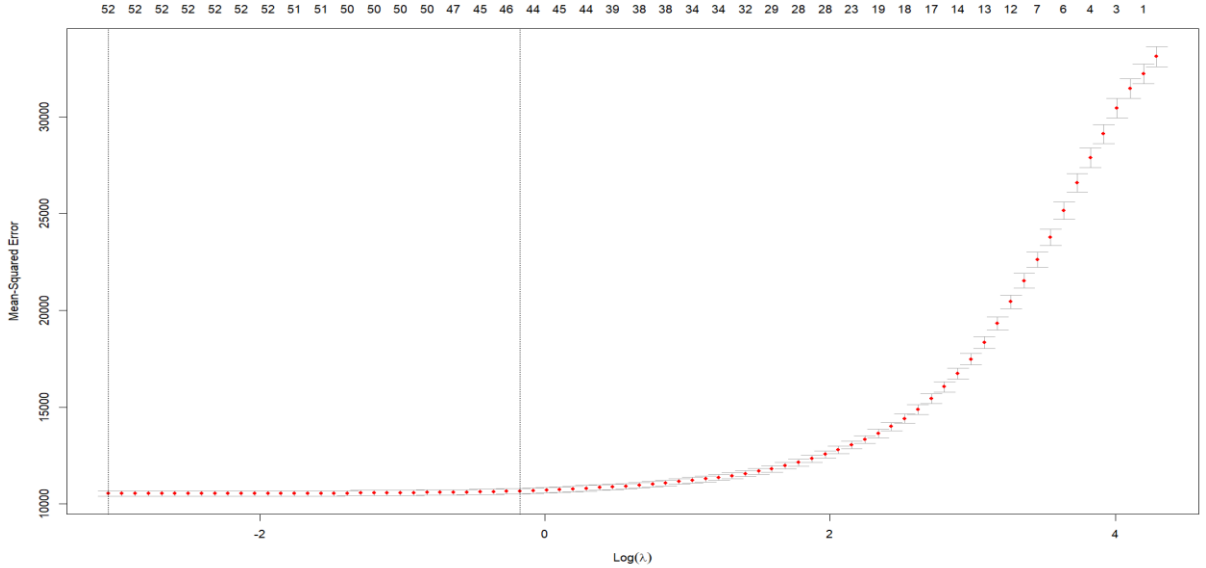
```
> lasso.mod=glmnet(X[trainIndex,],y[trainIndex],alpha=1,lambda=grid,nfolds=5)
> lasso.mod

Call:  glmnet(x = X[trainIndex, ], y = y[trainIndex], alpha = 1, lambda = grid,      nfolds = 5)

> cv.out.lasso

Call:  cv.glmnet(x = X[trainIndex, ], y = y[trainIndex], nfolds = 5,      alpha = 1)

Measure: Mean-Squared Error

    Lambda Index Measure    SE Nonzero
min  0.047    80   10561 126.5      52
1se  0.840    49   10679 139.3      45
```

```
> bestlam
[1] 0.04696182
```

```
> lasso.coef
(Intercept)     season2     season3     season4         yr1       mnth2       mnth3       mnth4       mnth5       mnth6       mnth7
189.4839155  16.8577497  13.6403741  28.6140927  42.6070787   0.5347265   3.4018016   0.9128913   4.8143983   0.7602159  -4.4177021
      mnth8       mnth9      mnth10      mnth11      mnth12         hr1         hr2         hr3         hr4         hr5         hr6
  1.4885979   8.3489765   4.1107508  -2.7242267  -1.8195189  -4.3940038  -6.1581060  -8.1785361  -8.7635333  -5.5658046   6.0917871
        hr7         hr8         hr9        hr10        hr11        hr12        hr13        hr14        hr15        hr16        hr17
 33.1206990  61.2071903  31.6025486  20.6256687  25.6801461  33.5438328  32.5368973  29.3447323  31.2373667  43.7272945  74.5796167
       hr18        hr19        hr20        hr21        hr22        hr23    holiday1    weekday1    weekday2    weekday3    weekday4
 68.1020591  46.3611803  30.4328204  20.5480397  13.1663565   5.4087410  -4.3636285   2.8973294   3.4436240   4.4173193   4.2494181
   weekday5    weekday6 weathersit2 weathersit3 weathersit4        temp       atemp         hum
   5.7814294   5.3450470  -4.4654339 -17.6923249  -0.7710976  23.3011725  22.0968532 -16.1323388
> lasso.coef[lasso.coef!=0]
(Intercept)     season2     season3     season4         yr1       mnth2       mnth3       mnth4       mnth5       mnth6       mnth7
189.4839155  16.8577497  13.6403741  28.6140927  42.6070787   0.5347265   3.4018016   0.9128913   4.8143983   0.7602159  -4.4177021
      mnth8       mnth9      mnth10      mnth11      mnth12         hr1         hr2         hr3         hr4         hr5         hr6
  1.4885979   8.3489765   4.1107508  -2.7242267  -1.8195189  -4.3940038  -6.1581060  -8.1785361  -8.7635333  -5.5658046   6.0917871
        hr7         hr8         hr9        hr10        hr11        hr12        hr13        hr14        hr15        hr16        hr17
 33.1206990  61.2071903  31.6025486  20.6256687  25.6801461  33.5438328  32.5368973  29.3447323  31.2373667  43.7272945  74.5796167
       hr18        hr19        hr20        hr21        hr22        hr23    holiday1    weekday1    weekday2    weekday3    weekday4
 68.1020591  46.3611803  30.4328204  20.5480397  13.1663565   5.4087410  -4.3636285   2.8973294   3.4436240   4.4173193   4.2494181
   weekday5    weekday6 weathersit2 weathersit3 weathersit4        temp       atemp         hum
   5.7814294   5.3450470  -4.4654339 -17.6923249  -0.7710976  23.3011725  22.0968532 -16.1323388
```

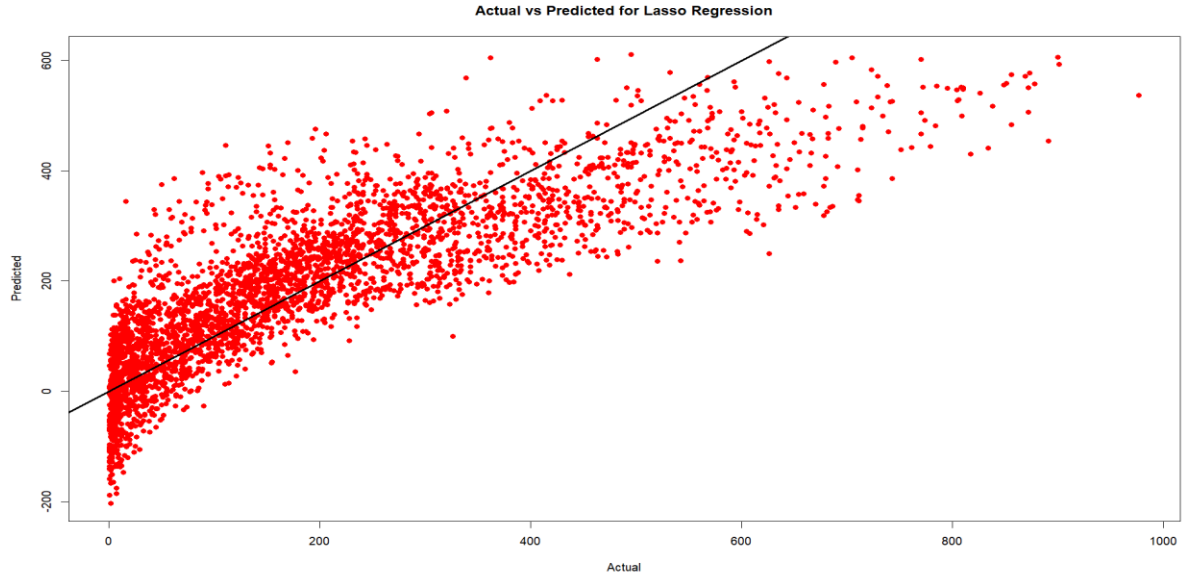
LASSO REGRESSION





Prediction with test data

```
> # Display the results
> cat("R-squared: ", r_squared_lr, "\n")
R-squared:  0.6929642
> cat("Mean Square Error (MsE): ", mse_lr, "\n")
Mean Square Error (MsE):  9840.154
> cat("Root Mean Square Error (RMSE): ", rmse_lr, "\n")
Root Mean Square Error (RMSE):  99.19755
```



RIDGE REGRESSION

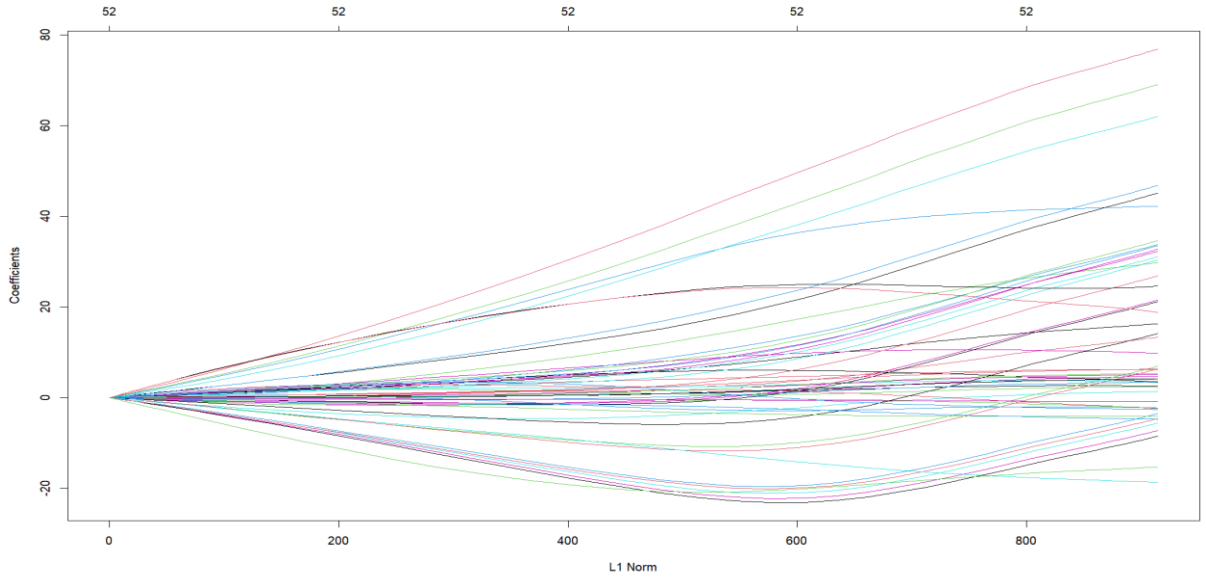
```
> ridge.mod=glmnet(X[trainIndex,],y[trainIndex],alpha=0,lambda=grid,nfold = 5)
> ridge.mod

Call:  glmnet(x = X[trainIndex, ], y = y[trainIndex], alpha = 0, lambda = grid,      nfold = 5)
```

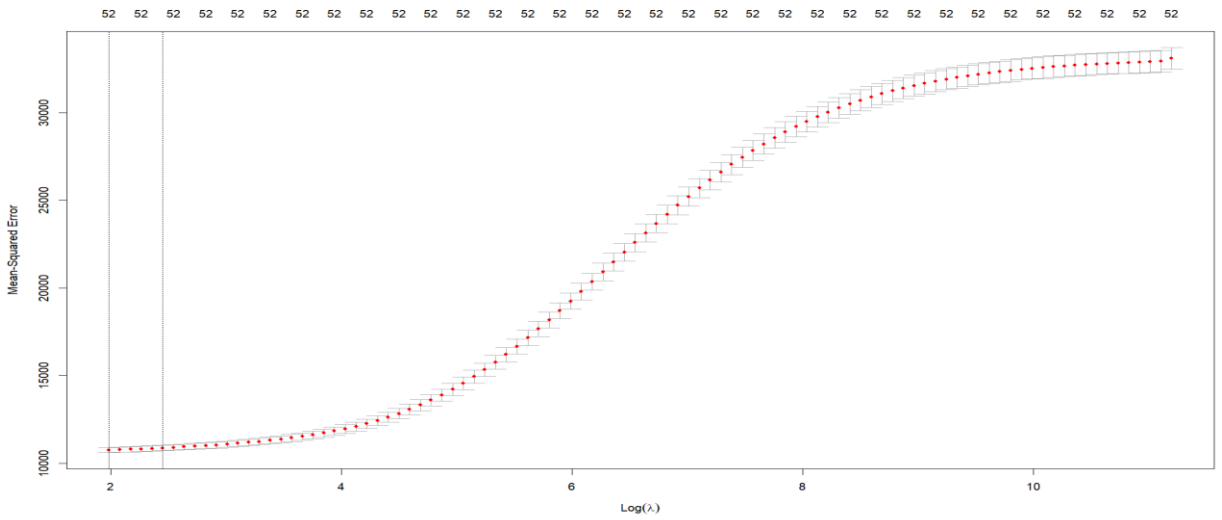
```
> cv.out

Call:  cv.glmnet(x = X[trainIndex, ], y = y[trainIndex], nfolds = 5,      alpha = 0)

Measure: Mean-Squared Error

    Lambda Index Measure    SE Nonzero
min  7.306   100   10752 146.1      52
1se 11.633    95   10871 157.6      52
```

```
> bestlam
[1] 7.305738
```

```
> ridge.coef
(Intercept)     season2     season3     season4         yr1       mnth2       mnth3       mnth4       mnth5       mnth6       mnth7       mnth8       mnth9
189.4839155  12.5550578   7.7016991  22.8651171  40.6945589  -0.4186437   2.9264633   1.8598086   5.6008711   1.4196220  -3.1470141   2.9491512   9.9927804
     mnth10      mnth11      mnth12          hr1         hr2         hr3         hr4         hr5         hr6         hr7         hr8         hr9         hr10
  6.2129840  -0.1525871  -0.2738258 -14.8754593 -16.4521741 -18.2137968 -18.7318176 -15.7770654  -4.5358967  21.3845958  48.3075871  19.6213550   8.8735587
        hr11        hr12        hr13        hr14        hr15        hr16        hr17        hr18        hr19        hr20        hr21        hr22        hr23
 13.5712367  20.9766971  19.8998662  16.7523456  18.5526797  30.5976738  60.3741305  54.2571669  33.4940456  18.2856256   8.9042161   1.8890558  -5.5148611
    holiday1    weekday1    weekday2    weekday3    weekday4    weekday5    weekday6 weathersit2 weathersit3 weathersit4        temp       atemp         hum
 -4.4042221   2.3121205   2.8247142   3.7550335   3.5338167   5.0127536   4.6124441  -3.4852771 -15.9546090  -0.6671070  24.2334388  23.9738209 -18.6403522
> ridge.coef[ridge.coef!=0]
(Intercept)     season2     season3     season4         yr1       mnth2       mnth3       mnth4       mnth5       mnth6       mnth7       mnth8       mnth9
189.4839155  12.5550578   7.7016991  22.8651171  40.6945589  -0.4186437   2.9264633   1.8598086   5.6008711   1.4196220  -3.1470141   2.9491512   9.9927804
     mnth10      mnth11      mnth12          hr1         hr2         hr3         hr4         hr5         hr6         hr7         hr8         hr9         hr10
  6.2129840  -0.1525871  -0.2738258 -14.8754593 -16.4521741 -18.2137968 -18.7318176 -15.7770654  -4.5358967  21.3845958  48.3075871  19.6213550   8.8735587
        hr11        hr12        hr13        hr14        hr15        hr16        hr17        hr18        hr19        hr20        hr21        hr22        hr23
 13.5712367  20.9766971  19.8998662  16.7523456  18.5526797  30.5976738  60.3741305  54.2571669  33.4940456  18.2856256   8.9042161   1.8890558  -5.5148611
    holiday1    weekday1    weekday2    weekday3    weekday4    weekday5    weekday6 weathersit2 weathersit3 weathersit4        temp       atemp         hum
 -4.4042221   2.3121205   2.8247142   3.7550335   3.5338167   5.0127536   4.6124441  -3.4852771 -15.9546090  -0.6671070  24.2334388  23.9738209 -18.6403522
```

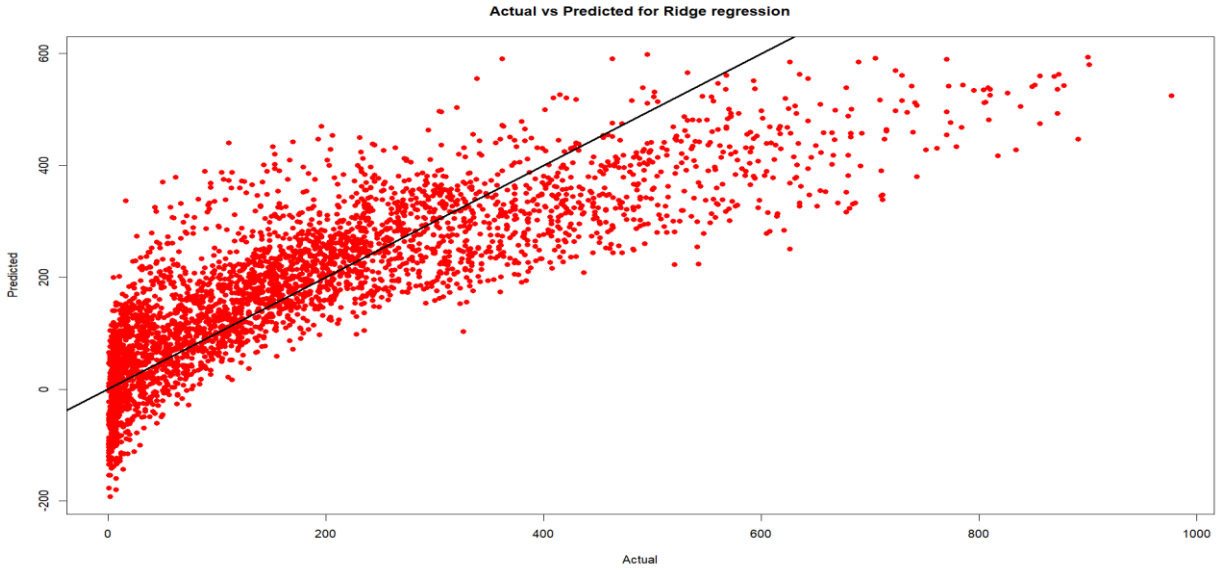
RIDGE REGRESSION





Prediction with test data

```
> cat("R-squared: ", r_squared_rr, "\n")
R-squared:  0.6877307
> cat("Mean Square Error (MsE): ", mse_rr, "\n")
Mean Square Error (MsE):  10007.88
> cat("Root Mean Square Error (RMSE): ", rmse_rr, "\n")
Root Mean Square Error (RMSE):  100.0394
```

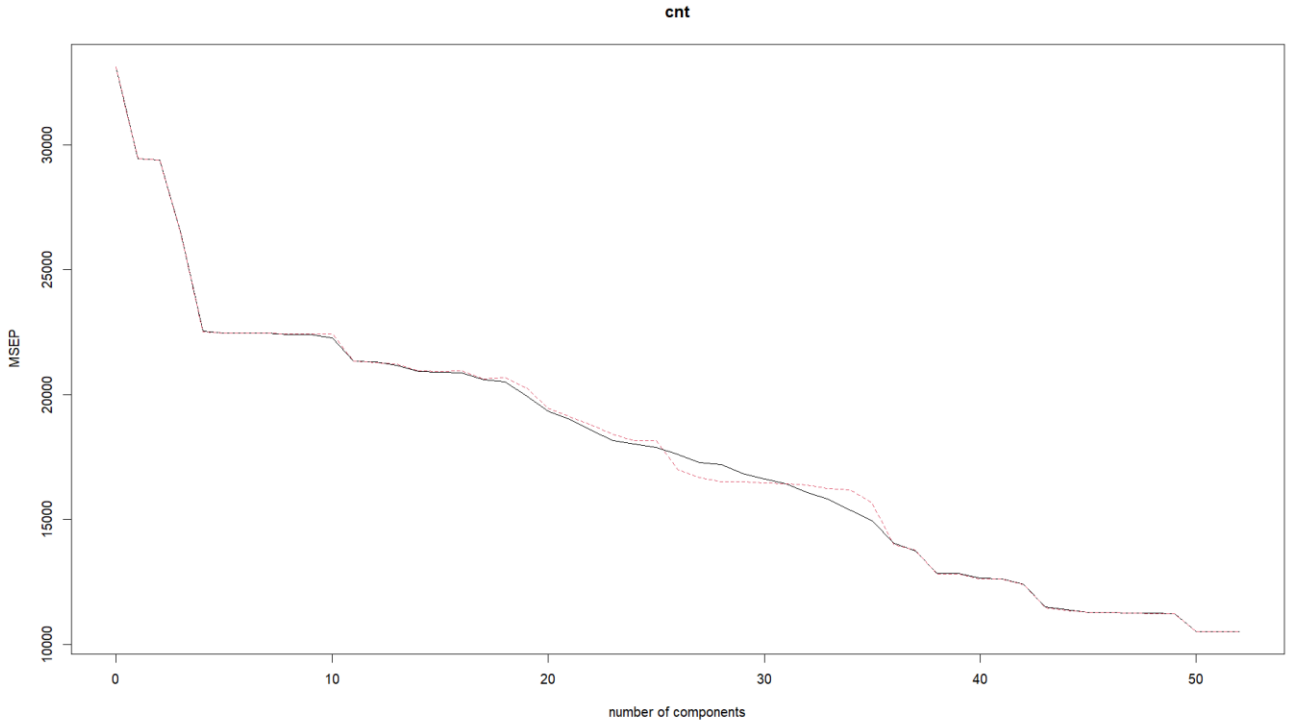


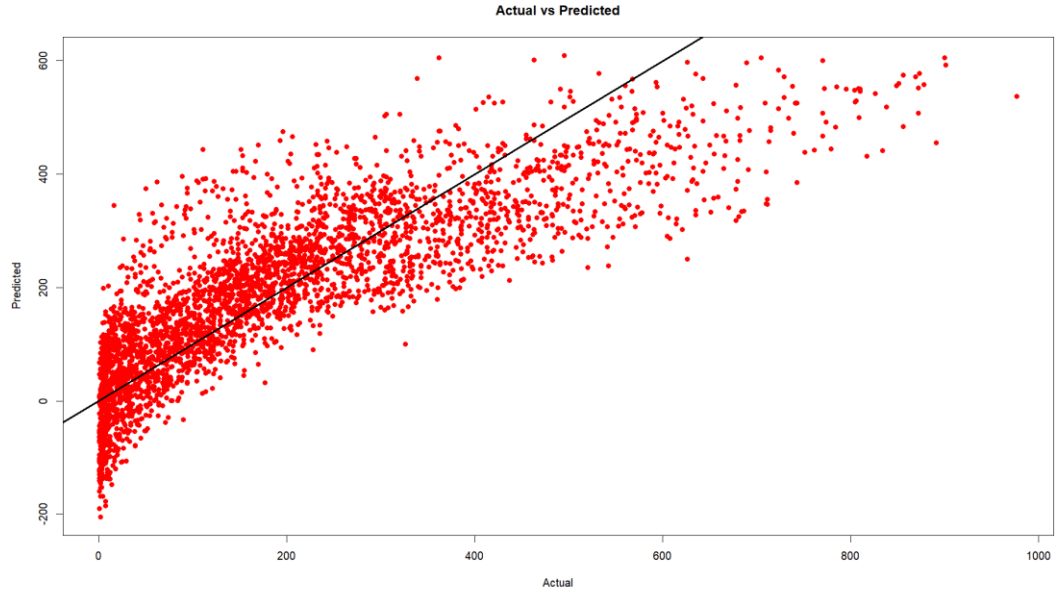
DIMENSION REDUCTION METHOD PRINCIPLE COMPONENT REGRESSION

PRINCIPLE COMPONENT REGRESSION

```
> pcr.fit
Principal component regression, fitted with the singular value decomposition algorithm.
Cross-validated using 10 random segments.
Call:
pcr(formula = cnt ~ ., data = dff, subset = trainIndex, scale = TRUE,     validation = "CV")
> summary(pcr.fit)
Data:   X dimension: 13902 52
        Y dimension: 13902 1
Fit method: svdpc
Number of components considered: 52
```

```
> cat("R-squared: ", r_squared_pca, "\n")
R-squared:  0.692995
> cat("Mean Square Error (MsE): ", mse_pca, "\n")
Mean Square Error (MsE):  9839.166
> cat("Root Mean Square Error (RMSE): ", rmse_pca, "\n")
Root Mean Square Error (RMSE):  99.19257
```

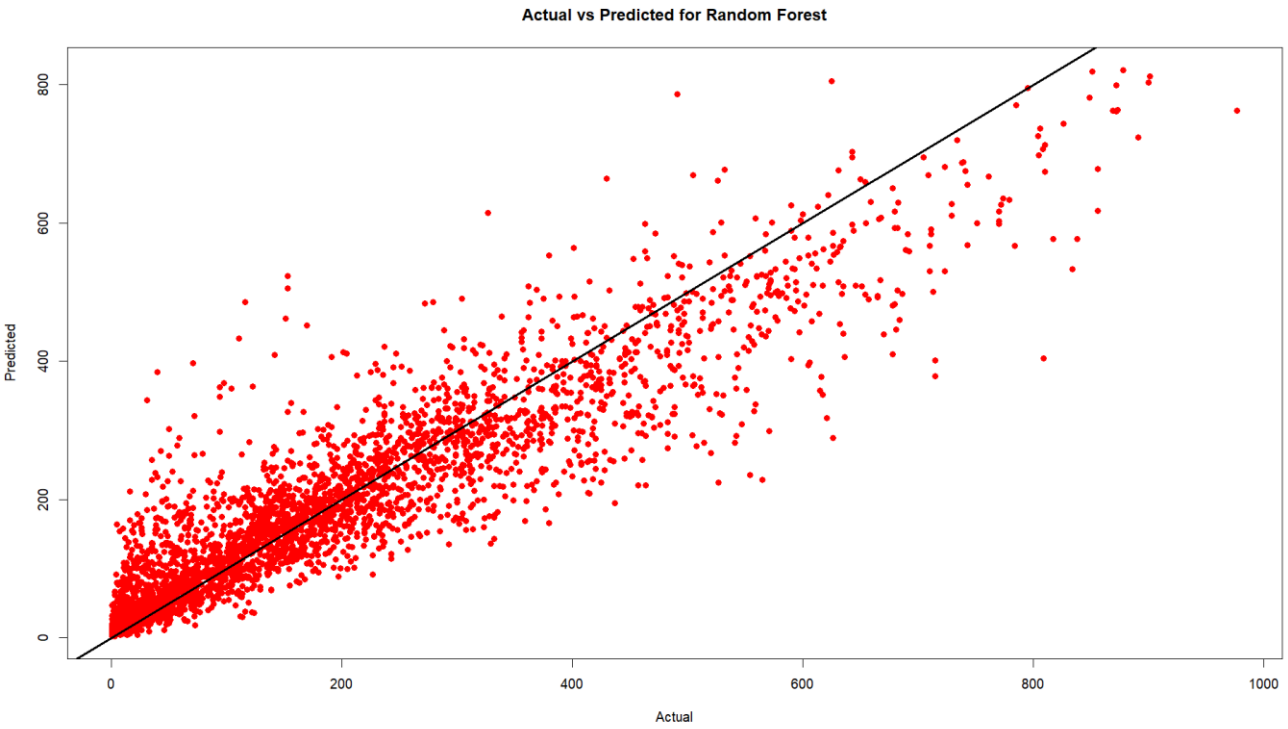




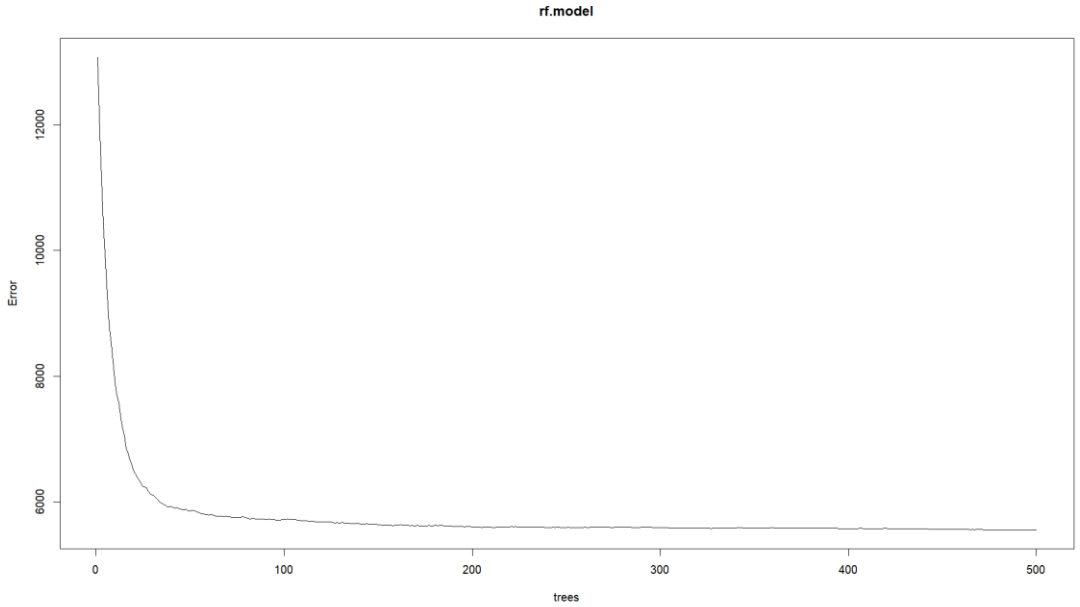
RANDOM FOREST

RANDOM FOREST

```
> rf.model

Call:
 randomForest(formula = cnt ~ ., data = dff, importance = TRUE,      subset = trainIndex)
               Type of random forest: regression
                     Number of trees: 500
No. of variables tried at each split: 17

          Mean of squared residuals: 5555.428
                    % Var explained: 83.22
```



Trees Vs Error



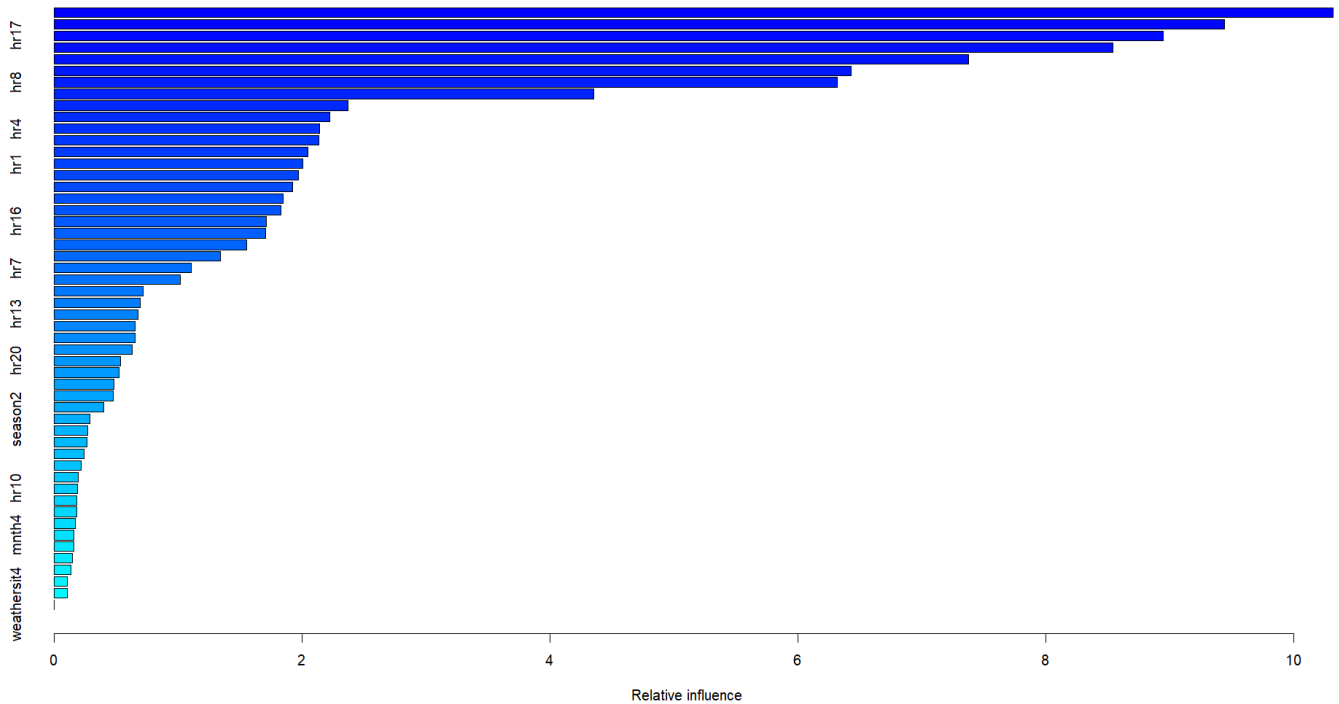

```
> cat("R-squared: ", r_squared_rf, "\n")
R-squared:  0.8443822
> cat("Mean Square Error (MsE): ", mse_rf, "\n")
Mean Square Error (MsE):  4987.375
> cat("Root Mean Square Error (RMSE): ", rmse_rf, "\n")
Root Mean Square Error (RMSE):  70.62135
```

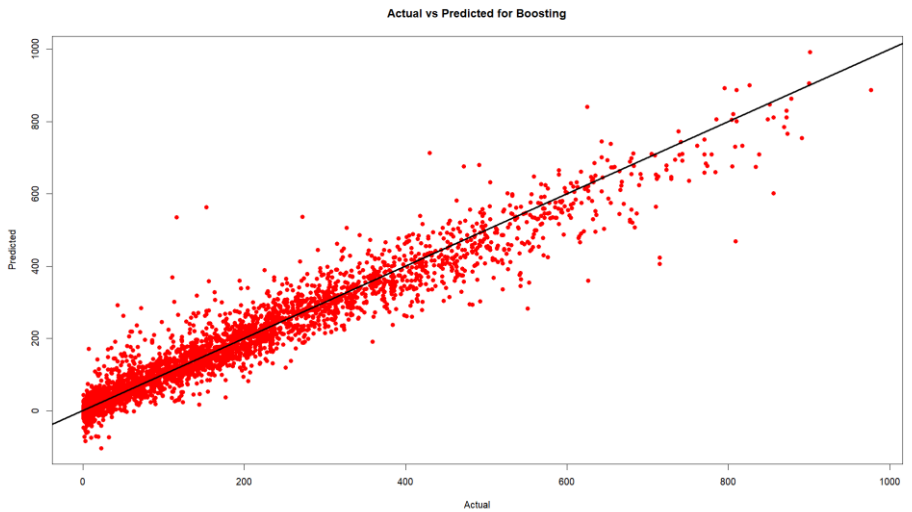
BOOSTING TECHNIQUE
GRADIENT BOOSTING MACHINE

GBM

```
> boost.model=gbm(cnt~.,data=dff[trainIndex,],distribution="gaussian",n.trees=5000,cv.folds = 5,interaction.depth=6)
> boost.model
gbm(formula = cnt ~ ., distribution = "gaussian", data = dff[trainIndex,
    ], n.trees = 5000, interaction.depth = 6, cv.folds = 5)
A gradient boosted model with gaussian loss function.
5000 iterations were performed.
The best cross-validation iteration was 4806.
There were 52 predictors of which 51 had non-zero influence.
```

```
> cat("R-squared: ", r_squared_bm, "\n")
R-squared:  0.9289999
> cat("Mean Square Error (MsE): ", mse_bm, "\n")
Mean Square Error (MsE):  2275.473
> cat("Root Mean Square Error (RMSE): ", rmse_bm, "\n")
Root Mean Square Error (RMSE):  47.70192
```

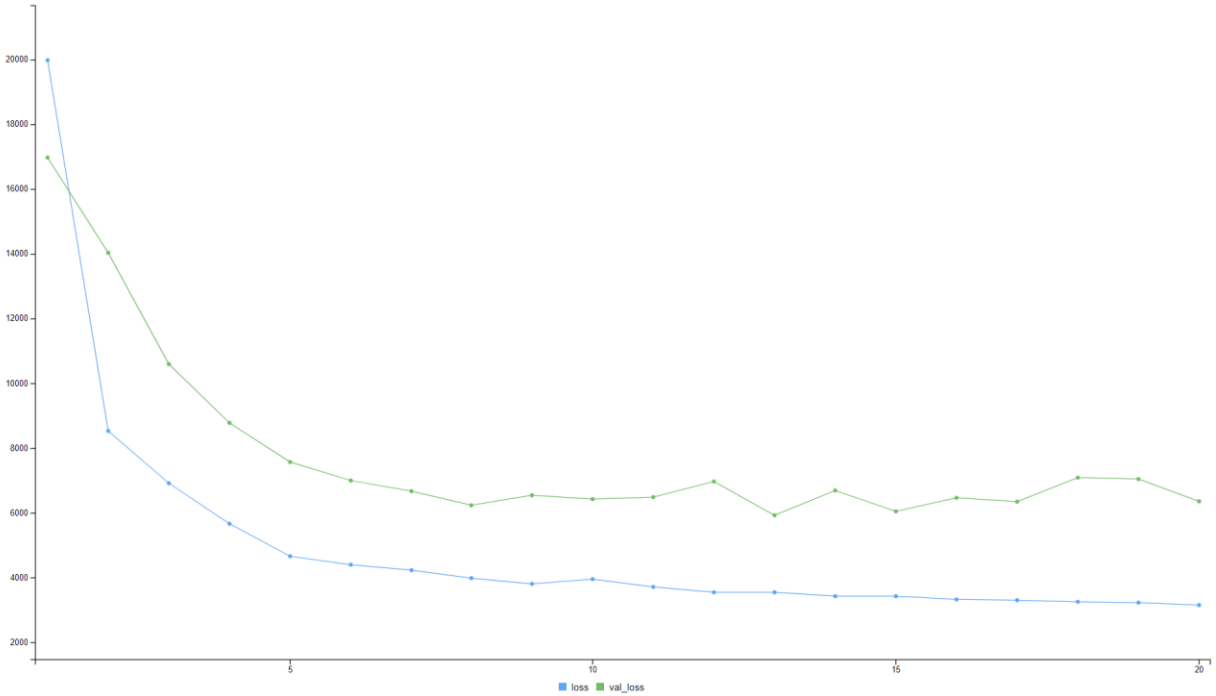




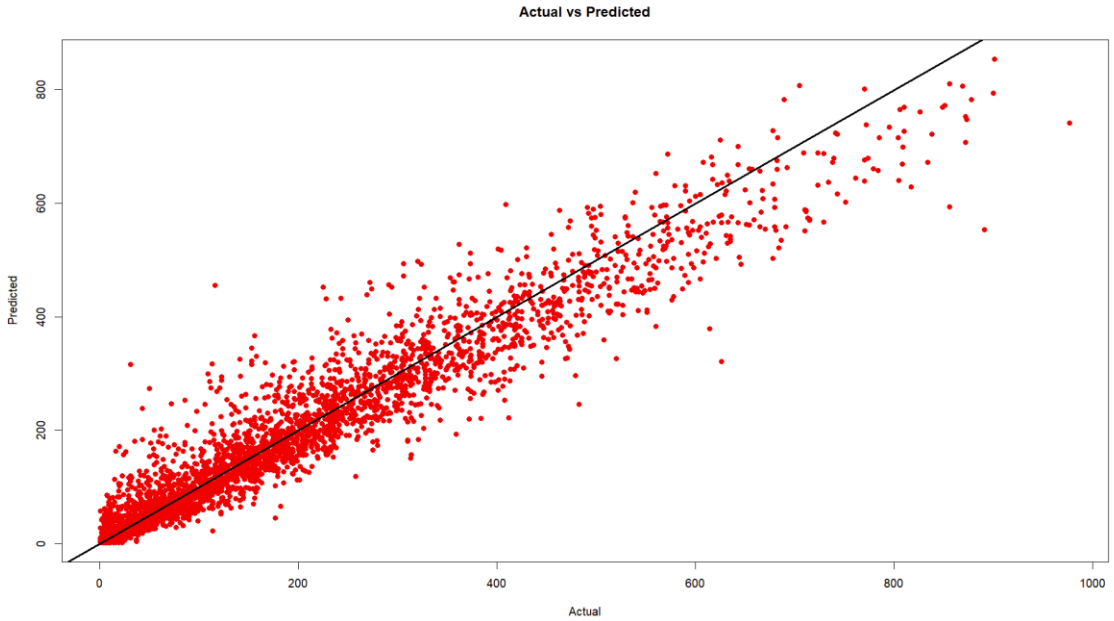
NEURAL NETWORK

NEURAL NETWORK

- A neural network is a computational model inspired by the brain's biological networks, using interconnected nodes or neurons to process and learn from data. It excels in pattern recognition and problem-solving in various AI and machine learning applications.




```
> model
Model: "sequential_17"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_69 (Dense)            (None, 128)               6784
 dropout_43 (Dropout)        (None, 128)               0
 dense_68 (Dense)            (None, 64)                8256
 dropout_42 (Dropout)        (None, 64)                0
 dense_67 (Dense)            (None, 32)                2080
 dropout_41 (Dropout)        (None, 32)                0
 dense_66 (Dense)            (None, 1)                 33
=================================================================
Total params: 17153 (67.00 KB)
Trainable params: 17153 (67.00 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

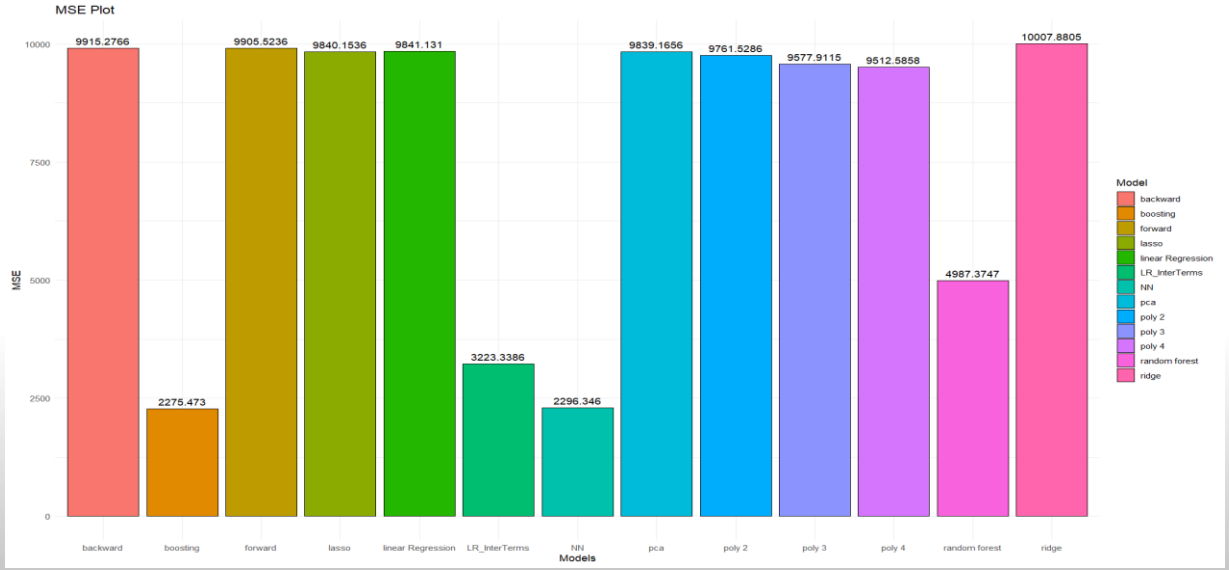



```
> npred[90:120]
 [1]  53.524258 100.834198  95.815201  54.177750  16.889029  14.479589   6.393068  59.633827  47.562016
[10]  34.576405  34.139233 159.828445 122.070030   4.273767  28.845617  53.216862  10.493546  88.245804
[19]  63.974056  37.307816  73.467369  37.573139  11.945144   8.971384  38.799953  68.332489 155.409195
[28]  52.531227  21.633905  43.107937  97.458145
> y_test[90:120]
 [1]  79 104 118  67  17   8   2  47  22  28  35 125 133   2  52  59   6 143  98  81  91  56  21   2  63  65
[27] 161  53  11  40  70
```

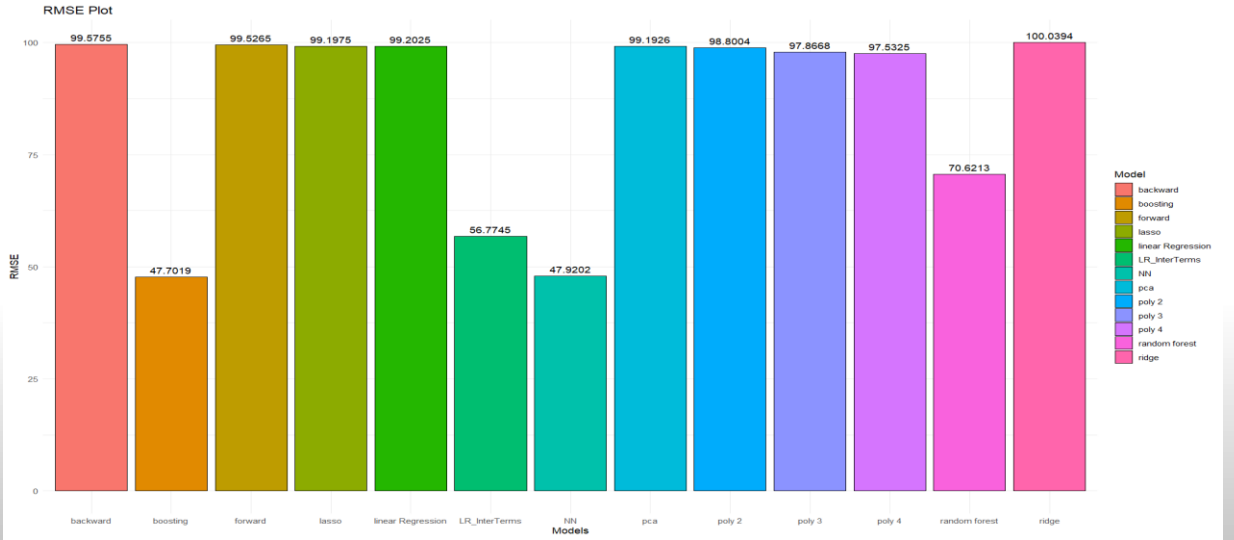
```
> cat("R-squared: ", r_squared_nn, "\n")
R-squared:  0.9231235
> cat("Mean Square Error (MsE): ", mse_nn, "\n")
Mean Square Error (MsE):  2463.806
> cat("Root Mean Square Error (RMSE): ", rmse_nn, "\n")
Root Mean Square Error (RMSE):  49.63674
>
```

NEURAL NETWORK

RESULTS

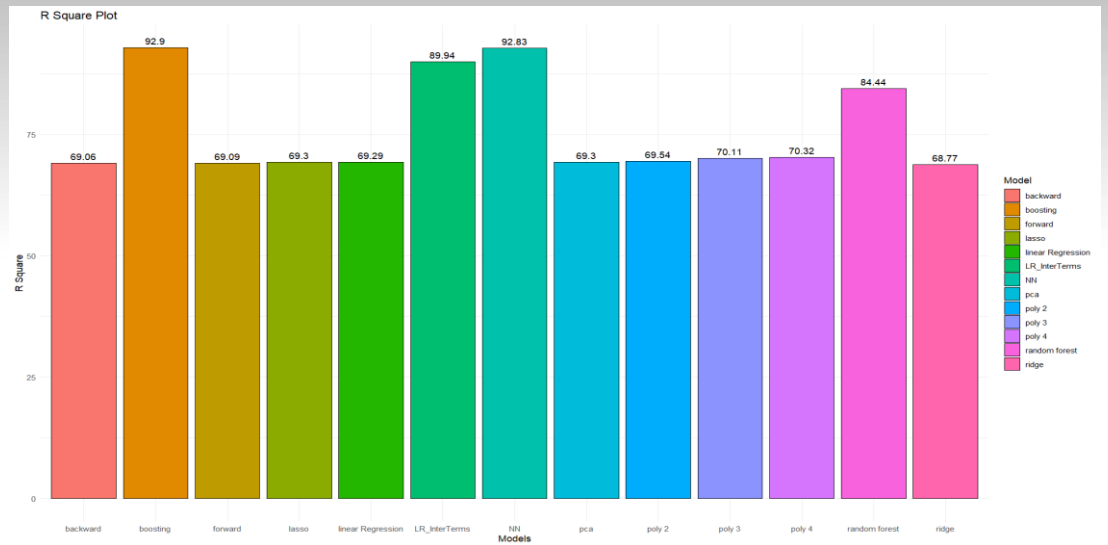


Mean Square Error (MSE)



Root Mean Square Error (RMSE)

R Square (R^2)



MODEL EVALUATION

Model	MSE	RMSE	R^2
Linear Regression	9841.131	99.20247	69.29337
Linear Regression – Poly 2	9761.529	98.80045	69.54175
Linear Regression – Poly 3	9577.911	97.8668	70.11467
Linear Regression – Poly 4	9512.586	97.53249	70.31851
Linear Regression – Interaction term	3223.339	56.77445	89.94243
Forward Sub Selection	9905.524	99.5265	69.09245
Backward Sub selection	9915.277	99.57548	69.06202
Lasso Regression	11595.847	107.68402	63.81824
Ridge Regression	9980.374	99.90182	68.8589
Random Forest	4987.375	70.62135	84.43822
Gradient Boosting Machine	2275.473	47.70192	92.89999
Principle Component Analysis	9839.166	99.19257	69.2995
Neural Network	2362.696	48.60757	92.62784

CONCLUSION

- The Gradient Boosting Machine (GBM) and Neural Network models outperformed others in predicting bike-sharing demand, with the GBM showing the lowest error rates and a high R^2 value, highlighting its precise predictions.
- The elapsed time for Neural Network is good when compared to GBM, RF, Linear Regression with interaction terms.
- While traditional models like Linear Regression lagged in performance, introducing interaction terms significantly improved results, underscoring the importance of feature interplay in predictive accuracy.
- The Random Forest algorithm also demonstrated solid predictive capabilities, suggesting it as a reliable alternative. Overall, the GBM and Neural Network are the preferred models for their superior prediction quality in bike-sharing demand forecasting.

FUTURE SCOPE

- In the future, our project aims to significantly expand its scope and impact. We plan to integrate real-time data to enhance prediction accuracy, and extend the model to encompass a wider range of urban transportation modes, providing a more comprehensive view of city mobility. Delving deeper into advanced machine learning techniques, we'll strive to refine our predictive capabilities, ensuring the model adapts to diverse urban environments globally.
- Additionally, by analyzing user behavior more closely, we aim to tailor the bike-sharing experience, increasing its efficiency and appeal. These steps are not just about advancing the project, but also about contributing to larger goals like reducing urban congestion and promoting sustainable transportation in cities around the world.