bikes.R

91863

2021-01-31

```
####Projet title :- Bike Renting using R####

#Problem statement :-
#The objective of this Case is to Predication of bike rental count on daily based
#on the environmental and seasonal settings.

# INDEPENDENT VARIABLE: "RENTED BIKE COUNT"

#### We will do following steps
#1.import the data set
#2.remove unnecessary columns
#3.missing value analysis
#4.ouliers analysis
#5.seasons wise monthly distributions count
#6.working day wise distribution counts
#7.Encoding the categorical findMethodSignatures()
#8.split the dataset into train and test dataset
#9.modeling the training dataset
#10.cross validation prediction
#11.model performance on test dataset
#12.model Evaluation metrics
#13.choosing best model for predicting bike rental count

#To remove previous outputs and files in Rstudio
rm(list=ls())
#read xlsx file
library(readxl)
bike=read_excel("seoul bike count.xlsx")
View(bike)

#remove unnecessary columns
bike=bike[-1]
dim(bike)
```

```
## [1] 8760   13
```

```
#check the missing values of the data
data.frame(colSums(is.na(bike)))
```

```
##                          colSums.is.na.bike..
## Rented Bike Count                           0
```

```
## Hour                                    0
## Temperature(°C)                         0
## Humidity(%)                             5
## Wind speed (m/s)                       37
## Visibility (10m)                       51
## Dew point temperature(°C)               0
## Solar Radiation (MJ/m2)                 0
## Rainfall(mm)                            0
## Snowfall (cm)                           0
## Seasons                                28
## Holiday                                 0
## Functioning Day                         0
```

```
#summary of data
summary(bike)
```

```
##  Rented Bike Count      Hour       Temperature(°C)   Humidity(%)
##  Min.   :   0.0    Min.   : 0.00   Min.   :-17.80   Min.   : 0.00
##  1st Qu.: 191.0    1st Qu.: 5.75   1st Qu.:  3.50   1st Qu.:42.00
##  Median : 504.5    Median :11.50   Median : 13.70   Median :57.00
##  Mean   : 704.6    Mean   :11.50   Mean   : 12.88   Mean   :58.23
##  3rd Qu.:1065.2    3rd Qu.:17.25   3rd Qu.: 22.50   3rd Qu.:74.00
##  Max.   :3556.0    Max.   :23.00   Max.   : 39.40   Max.   :98.00
##                                                     NA's   :5
##  Wind speed (m/s) Visibility (10m) Dew point temperature(°C)
##  Min.   :0.000    Min.   :  27     Min.   :-30.600
##  1st Qu.:0.900    1st Qu.: 936     1st Qu.: -4.700
##  Median :1.500    Median :1693     Median :  5.100
##  Mean   :1.727    Mean   :1435     Mean   :  4.074
##  3rd Qu.:2.300    3rd Qu.:2000     3rd Qu.: 14.800
##  Max.   :7.400    Max.   :2000     Max.   : 27.200
##  NA's   :37       NA's   :51
##  Solar Radiation (MJ/m2)  Rainfall(mm)     Snowfall (cm)       Seasons
##  Min.   :0.0000          Min.   : 0.0000   Min.   :0.00000   Length:8760
##  1st Qu.:0.0000          1st Qu.: 0.0000   1st Qu.:0.00000   Class :character
##  Median :0.0100          Median : 0.0000   Median :0.00000   Mode  :character
##  Mean   :0.5691          Mean   : 0.1487   Mean   :0.07507
##  3rd Qu.:0.9300          3rd Qu.: 0.0000   3rd Qu.:0.00000
##  Max.   :3.5200          Max.   :35.0000   Max.   :8.80000
##
##    Holiday          Functioning Day
##  Length:8760        Length:8760
##  Class :character   Class :character
##  Mode  :character   Mode  :character
##
##
##
##
```

```
#count analysis of categorical data
table(bike$Seasons)
```

```
##
```

```
## Autumn Spring Summer Winter
##   2171   2202   2199   2160
```

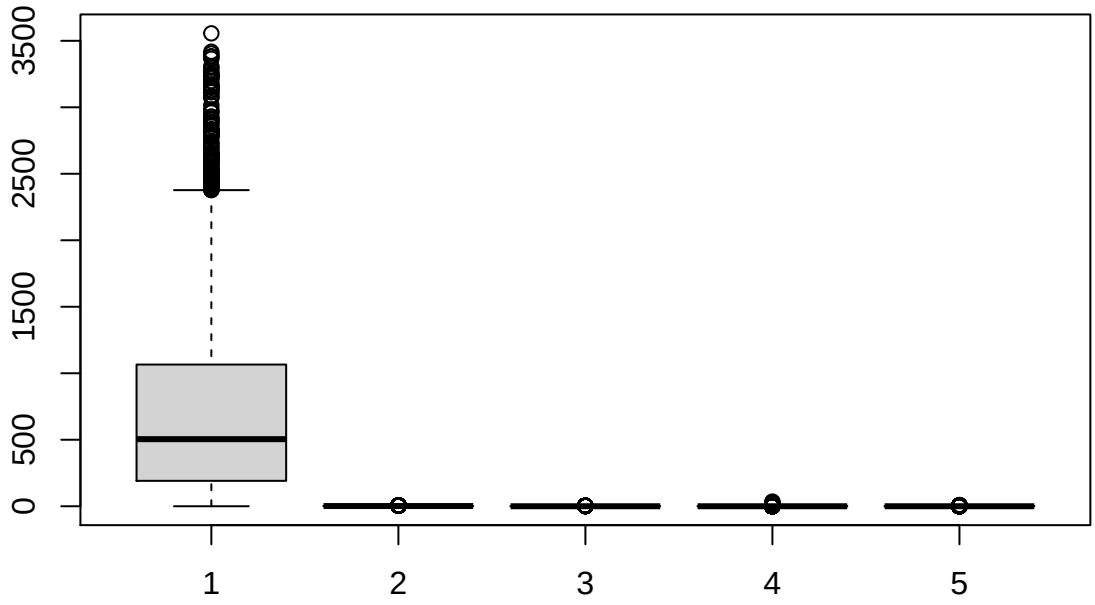
```
#str of the data
str(bike)
```

```
## tibble [8,760 x 13] (S3: tbl_df/tbl/data.frame)
##  $ Rented Bike Count        : num [1:8760] 254 204 173 107 78 100 181 460 930 490 ...
##  $ Hour                     : num [1:8760] 0 1 2 3 4 5 6 7 8 9 ...
##  $ Temperature(°C)          : num [1:8760] -5.2 -5.5 -6 -6.2 -6 -6.4 -6.6 -7.4 -7.6 -6.5 ...
##  $ Humidity(%)              : num [1:8760] 37 38 39 40 36 37 35 38 37 27 ...
##  $ Wind speed (m/s)         : num [1:8760] 2.2 0.8 1 0.9 2.3 1.5 1.3 0.9 1.1 0.5 ...
##  $ Visibility (10m)         : num [1:8760] 2000 2000 2000 2000 2000 ...
##  $ Dew point temperature(°C): num [1:8760] -17.6 -17.6 -17.7 -17.6 -18.6 -18.7 -19.5 -19.3 -19.8 -22
##  $ Solar Radiation (MJ/m2)  : num [1:8760] 0 0 0 0 0 0 0 0 0.01 0.23 ...
##  $ Rainfall(mm)             : num [1:8760] 0 0 0 0 0 0 0 0 0 0 ...
##  $ Snowfall (cm)            : num [1:8760] 0 0 0 0 0 0 0 0 0 0 ...
##  $ Seasons                  : chr [1:8760] "Winter" "Winter" "Winter" "Winter" ...
##  $ Holiday                  : chr [1:8760] "No Holiday" "No Holiday" "No Holiday" "No Holiday" ...
##  $ Functioning Day          : chr [1:8760] "Yes" "Yes" "Yes" "Yes" ...
```

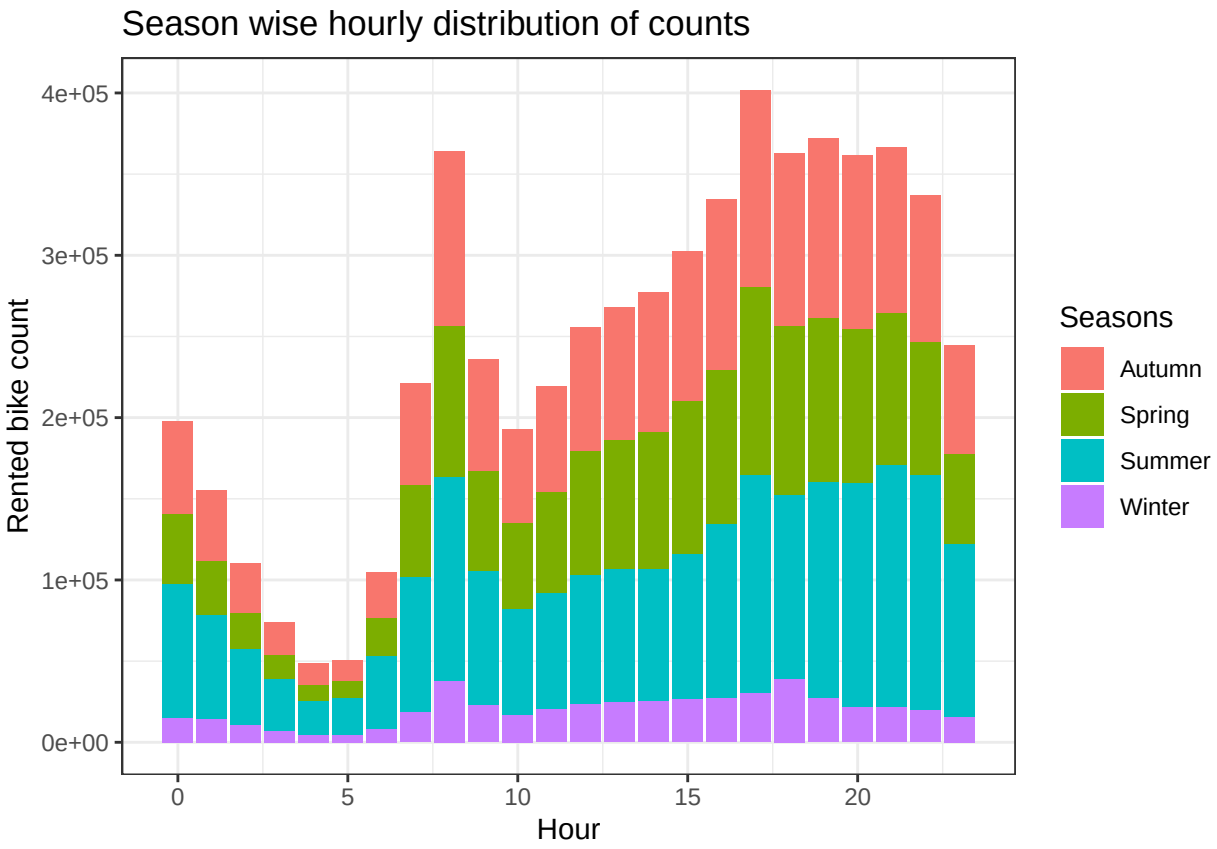
```
#dimensions of data
dim(bike)
```

```
## [1] 8760   13
```

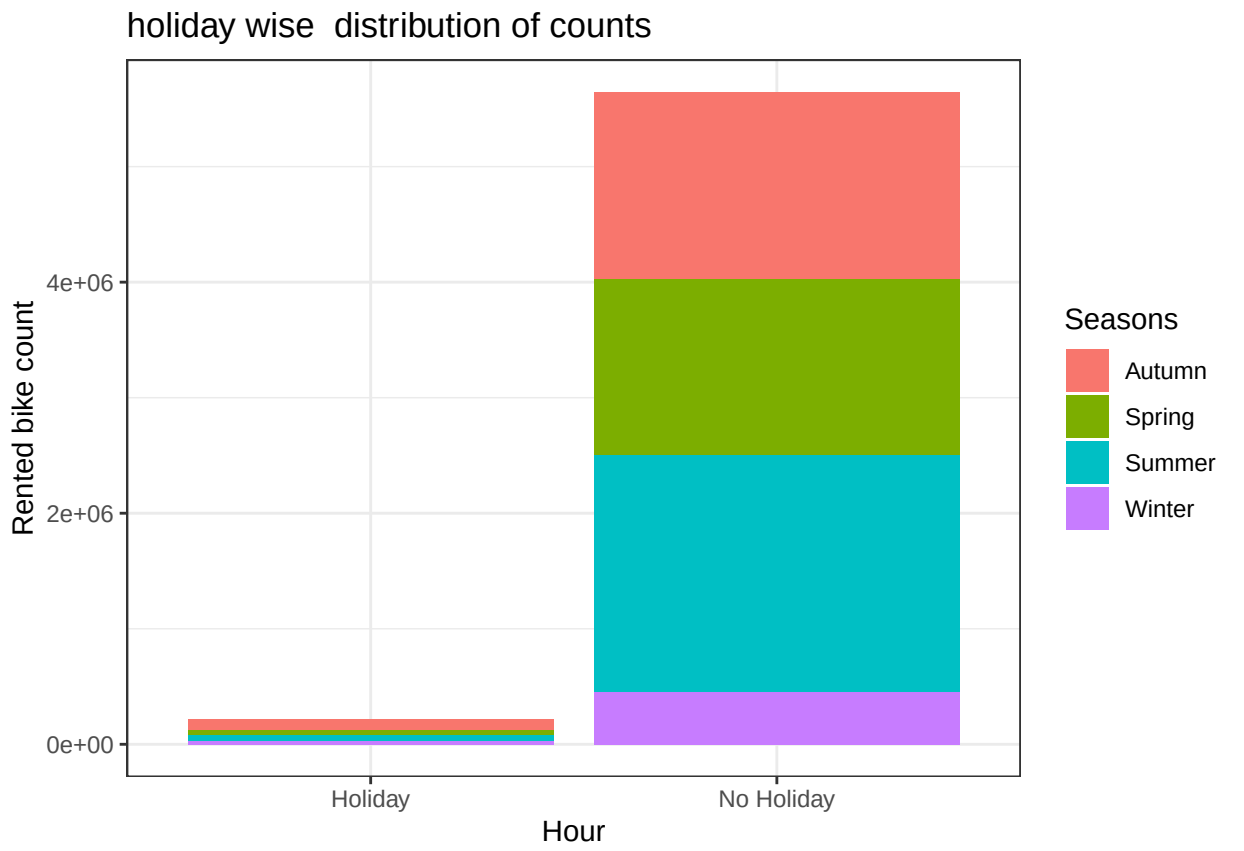
```
#missing values numerical data replace with mean
bike$`Humidity(%)`[is.na(bike$`Humidity(%)`)]=mean(bike$`Humidity(%)`,na.rm=T)
bike$`Wind speed (m/s)`[is.na(bike$`Wind speed (m/s)`)]=mean(bike$`Wind speed (m/s)`,na.rm=T)
bike$`Visibility (10m)`[is.na(bike$`Visibility (10m)`)]=mean(bike$`Visibility (10m)`,na.rm=T)

#missing values categorical data replace with mode
bike$Seasons[is.na(bike$Seasons)]="Spring"

#check the outliers using bixplots
boxplot(bike$`Rented Bike Count`,bike$`Wind speed (m/s)`, bike$`Solar Radiation (MJ/m2)`,
        bike$`Rainfall(mm)`,bike$`Snowfall (cm)`)

#outliers replace with mean
Outlier <- function(x){
  qnt <- quantile(x, probs=c(.25, .75), na.rm = T)
  H <- 1.5 * IQR(x)
  x[x < (qnt[1] - H)] <- mean(x)
  x[x > (qnt[2] + H)] <- mean(x)
  return(x)
}
bike$`Rented Bike Count`=Outlier(bike$`Rented Bike Count`)
bike$`Wind speed (m/s)`=Outlier(bike$`Wind speed (m/s)`)
bike$`Solar Radiation (MJ/m2)`=Outlier(bike$`Solar Radiation (MJ/m2)`)
bike$`Rainfall(mm)`=Outlier(bike$`Rainfall(mm)`)
bike$`Snowfall (cm)`=Outlier(bike$`Snowfall (cm)`)

library(ggplot2)
```




```
#dev.off()
#column plot for season wise monthly distribution of counts
ggplot(bike,aes(x=Hour,y=`Rented Bike Count`,
                fill=Seasons))+theme_bw()+geom_col()+labs(x='Hour',
          y='Rented bike count',title='Season wise hourly distribution of counts')
```



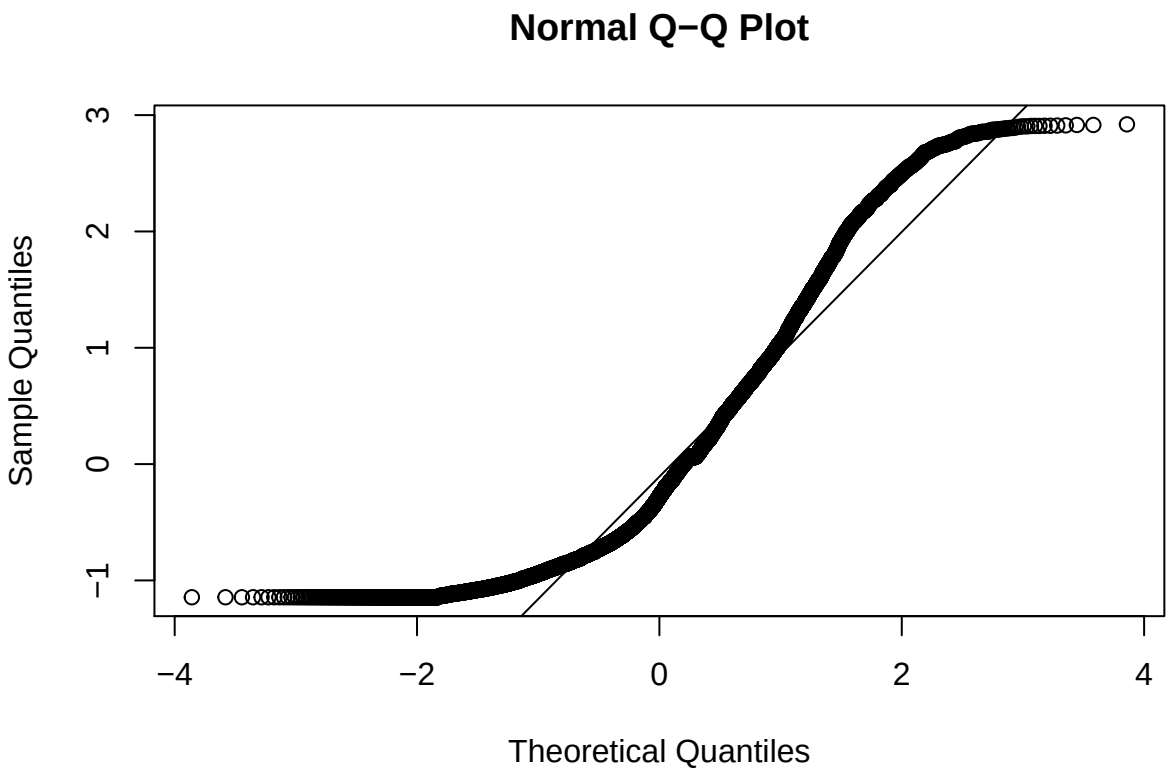

```
#OBSERVED:From the above plots, we can observed that increasing the bike rental count in spring and
#summer season and then decreasing the bike rental count in Autumn and winter season.
#Column plot for holiday wise distribution of counts
ggplot(bike,aes(x=Holiday,y=`Rented Bike Count`,
                fill=Seasons))+theme_bw()+geom_col()+labs(x='Hour', y='Rented bike count',
                                    title='holiday wise  distribution of counts')
```




```
#OBSERVED:From the above bar plot, we can observed that during no holiday the bike rental
#counts is highest compared to during holiday for different seasons.

#label encoding features
bike$Holiday=as.numeric(factor(bike$Holiday))-1
bike$`Functioning Day`=as.numeric(factor(bike$`Functioning Day`))-1

library(fastDummies)
bike1=fastDummies::dummy_cols(bike$Seasons,remove_first_dummy = F)
bike1=bike1[-1]
bike2=scale(bike[c(1,4,6)])
bike3=bike[c(2,3,5,7,8,9,10,12,13)]
bikee=cbind(bike3,bike2,bike1)
View(bikee)
dim(bikee)
```

```
## [1] 8760   16
```

```
#Quintle-Quintle line
qqnorm(bikee$`Rented Bike Count`)
qqline(bikee$`Rented Bike Count`)
```




```
#Split the dataset based on simple random resampling
train_index<-sample(1:nrow(bikee),0.7*nrow(bikee))
train_data<-bikee[train_index,]
test_data<-bikee[-train_index,]
dim(train_data)
```

```
## [1] 6132   16
```

```
dim(test_data)
```

```
## [1] 2628   16
```

```
############ Modelling the training dataset #########


######## 1.linear regression model #########
#Set seed to reproduce the results of random sampling
set.seed(672)
#train the lm model
lr_model=lm(train_data$`Rented Bike Count`~.,train_data[,c(-10)])
#Summary of the model
summary(lr_model)
```

```
##
```

```
## Call:
## lm(formula = train_data$`Rented Bike Count` ~ ., data = train_data[,
##     c(-10)])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.63242 -0.43178 -0.08959  0.32437  2.86021
##
## Coefficients: (1 not defined because of singularities)
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  -2.797444   0.082966 -33.718  < 2e-16 ***
## Hour                          0.041710   0.001319  31.611  < 2e-16 ***
## `Temperature(°C)`             0.025036   0.006579   3.805 0.000143 ***
## `Wind speed (m/s)`            0.016504   0.010153   1.626 0.104099
## `Dew point temperature(°C)`   0.012387   0.007011   1.767 0.077290 .
## `Solar Radiation (MJ/m2)`    -0.011188   0.017096  -0.654 0.512858
## `Rainfall(mm)`               -5.613449   0.263260 -21.323  < 2e-16 ***
## `Snowfall (cm)`               1.530297   0.566845   2.700 0.006960 **
## Holiday                       0.147837   0.039056   3.785 0.000155 ***
## `Functioning Day`             1.527494   0.048480  31.508  < 2e-16 ***
## `Humidity(%)`                -0.200643   0.039651  -5.060 4.31e-07 ***
## `Visibility (10m)`            0.031656   0.010821   2.925 0.003452 **
## .data_Autumn                  0.621736   0.036012  17.265  < 2e-16 ***
## .data_Spring                  0.424359   0.034175  12.417  < 2e-16 ***
## .data_Summer                  0.341084   0.051552   6.616 4.00e-11 ***
## .data_Winter                        NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6593 on 6117 degrees of freedom
## Multiple R-squared:  0.5611, Adjusted R-squared:  0.5601
## F-statistic: 558.5 on 14 and 6117 DF,  p-value: < 2.2e-16
```

```
#we will be using the caret package for crossvalidation.function named "trainControl".
#method="CV" (used for crossvalidation)
#number=5 (means 5 fold crossvalidation)
#classProbs=T (model will save the prediction for each class)
#"train" is a function available in caret pakage
#Cross validation resampling method
#To ignore warning messages
options(warn=-1)
library(caret)
```

```
## Loading required package: lattice
```

```
train.control=trainControl(method="CV",number=5,savePrediction=T,classProbs=T)
#Cross validation prediction
CV_predict=train(`Rented Bike Count`~.,data=train_data,method='lm',trControl=train.control)
#Summary of cross validation prediction
summary(CV_predict)
```

```
##
## Call:
```

```
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.63242 -0.43178 -0.08959  0.32437  2.86021
##
## Coefficients: (1 not defined because of singularities)
##                                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)                       -2.797444   0.082966 -33.718  < 2e-16 ***
## Hour                               0.041710   0.001319  31.611  < 2e-16 ***
## `\\`Temperature(°C)\\``            0.025036   0.006579   3.805 0.000143 ***
## `\\`Wind speed (m/s)\\``           0.016504   0.010153   1.626 0.104099
## `\\`Dew point temperature(°C)\\``  0.012387   0.007011   1.767 0.077290 .
## `\\`Solar Radiation (MJ/m2)\\``   -0.011188   0.017096  -0.654 0.512858
## `\\`Rainfall(mm)\\``              -5.613449   0.263260 -21.323  < 2e-16 ***
## `\\`Snowfall (cm)\\``              1.530297   0.566845   2.700 0.006960 **
## Holiday                            0.147837   0.039056   3.785 0.000155 ***
## `\\`Functioning Day\\``            1.527494   0.048480  31.508  < 2e-16 ***
## `\\`Humidity(%)\\``               -0.200643   0.039651  -5.060 4.31e-07 ***
## `\\`Visibility (10m)\\``           0.031656   0.010821   2.925 0.003452 **
## .data_Autumn                       0.621736   0.036012  17.265  < 2e-16 ***
## .data_Spring                       0.424359   0.034175  12.417  < 2e-16 ***
## .data_Summer                       0.341084   0.051552   6.616 4.00e-11 ***
## .data_Winter                             NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6593 on 6117 degrees of freedom
## Multiple R-squared:  0.5611, Adjusted R-squared:  0.5601
## F-statistic: 558.5 on 14 and 6117 DF,  p-value: < 2.2e-16
```

```
#OBSERVED:The adjusted R-squared or coefficient of determination is 0.548 on  cross validation ,
#it means that predictor is only able to predict 54% of the variance in the target
#variable which is contributed by independent variables.



######## 2.knn ##########

knn_model=train(`Rented Bike Count`~.,data=train_data,method="knn",trControl=train.control)#,preProcess
knn_model
```

```
## k-Nearest Neighbors
##
## 6132 samples
##   15 predictor
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 4906, 4905, 4905, 4906, 4906
## Resampling results across tuning parameters:
##
##   k  RMSE       Rsquared   MAE
##   5  0.5705102  0.6726163  0.3595746
```

```
##   7  0.5674463  0.6749478  0.3621277
##   9  0.5705005  0.6712219  0.3665173
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was k = 7.
```

```
######## 3.random forest ######
rf_model=train(`Rented Bike Count`~.,data=train_data,method="rf",trControl=train.control)#,preProcess=c
rf_model
```

```
## Random Forest
##
## 6132 samples
##   15 predictor
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 4905, 4906, 4906, 4906, 4905
## Resampling results across tuning parameters:
##
##   mtry  RMSE       Rsquared   MAE
##    2    0.5241870  0.7497423  0.3720326
##    8    0.4300232  0.8129279  0.2668379
##   15    0.4334589  0.8096672  0.2670616
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was mtry = 8.
```

```
###### Final model for predicting the bike rental count on daily basis
#OBSERVED:When we compare the root mean squared error and mean absolute error of all 3 models,
#the random forest model has less root mean squared error and mean absolute error.
#So, finally random forest model is best for predicting the bike rental count on daily basis.

predict=predict(rf_model,test_data)
tab=table(predict=predict,actual=test_data$`Rented Bike Count`)
View(head(tab))
```