DS2 Final Project

Wanxin Qi, Lesi He, Ke Xu

```
library(tidyverse)
library(lubridate)
library(caret)
library(ggplot2)
library(corrplot)
library(vip)
library(rpart.plot)
library(ranger)
library(GGally)
library(pdp)

knitr::opts_chunk$set(echo = TRUE, message = FALSE, warning = FALSE)

theme_set(theme_minimal() + theme(legend.position = "bottom"))
options(
  ggplot2.continuous.colour = "viridis",
  ggplot2.continuous.fill = "viridis")
scale_colour_discrete = scale_colour_viridis_d
scale_fill_discrete = scale_fill_viridis_d
```

Data

```
bike = read.csv("./data/SeoulBikeData.csv", check.names = F)

# Missing value in the dataset
sum(is.na(bike))
```

```
## [1] 0
```

```
# All the 0 hourly rented bike count are non-functioning day
bike %>%
  janitor::clean_names() %>%
  filter(rented_bike_count == 0) %>%
  count(functioning_day)
```

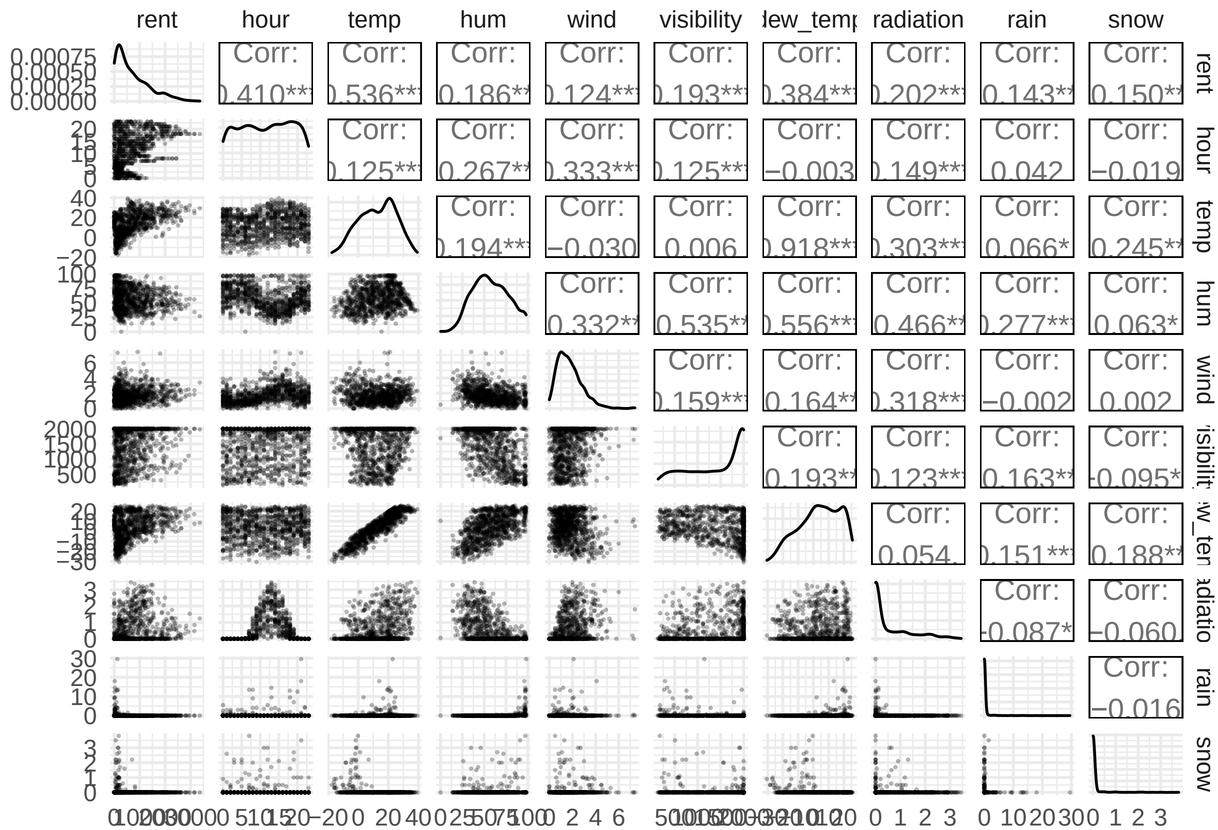
```
##   functioning_day   n
## 1              No 295
```

```
# tidy
bike = bike %>%
```

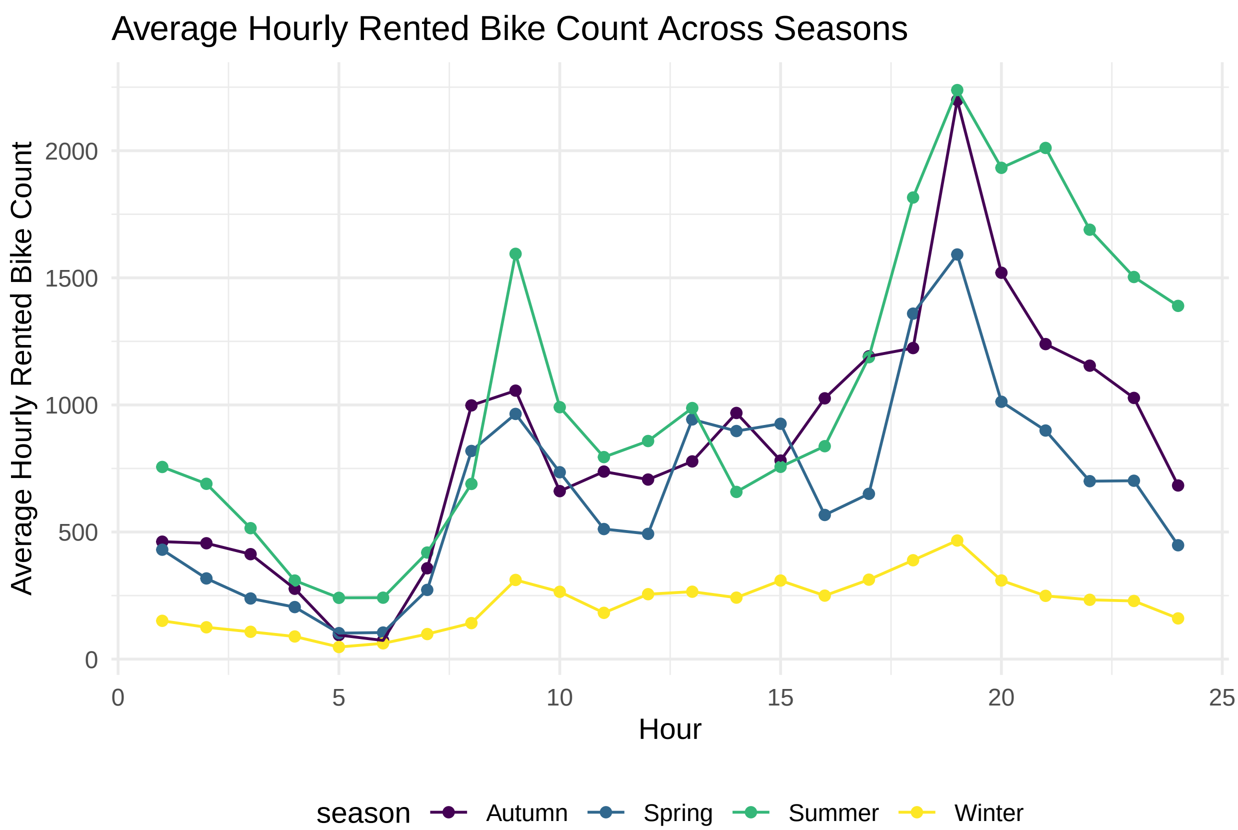
```r
  janitor::clean_names() %>%
  mutate(
    date = dmy(date),
    week = weekdays(date, abbreviate = TRUE),
    week = factor(week, levels = c("Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun")),
    rent = as.numeric(rented_bike_count),
    hour = as.numeric(hour),
    temp = as.numeric(temperature_c),
    hum = as.numeric(humidity_percent),
    wind = as.numeric(wind_speed_m_s),
    visibility = as.numeric(visibility_10m),
    dew_temp = as.numeric(dew_point_temperature_c),
    radiation = as.numeric(solar_radiation_mj_m2),
    rain = as.numeric(rainfall_mm),
    snow = as.numeric(snowfall_cm),
    season = as.factor(seasons),
    holiday = as.factor(ifelse(holiday == "No Holiday", "No", "Yes")),
    func = as.factor(functioning_day)
  ) %>%
  select(rent, hour, temp, hum, wind, visibility, dew_temp, radiation,
         rain, snow, season, week, holiday, func)

# Dataset of the research
set.seed(2022)
bike = bike[sample(nrow(bike), 1000),]

# Partition
set.seed(2)
trainRows = createDataPartition(y = bike$rent, p = 0.8, list = FALSE)
trainData = bike[trainRows,]
testData = bike[-trainRows,]

train_x = model.matrix(rent ~ ., bike)[trainRows, -1]
train_y = bike$rent[trainRows]
test_x = model.matrix(rent ~ ., bike)[-trainRows, -1]
test_y = bike$rent[-trainRows]
```

EDA

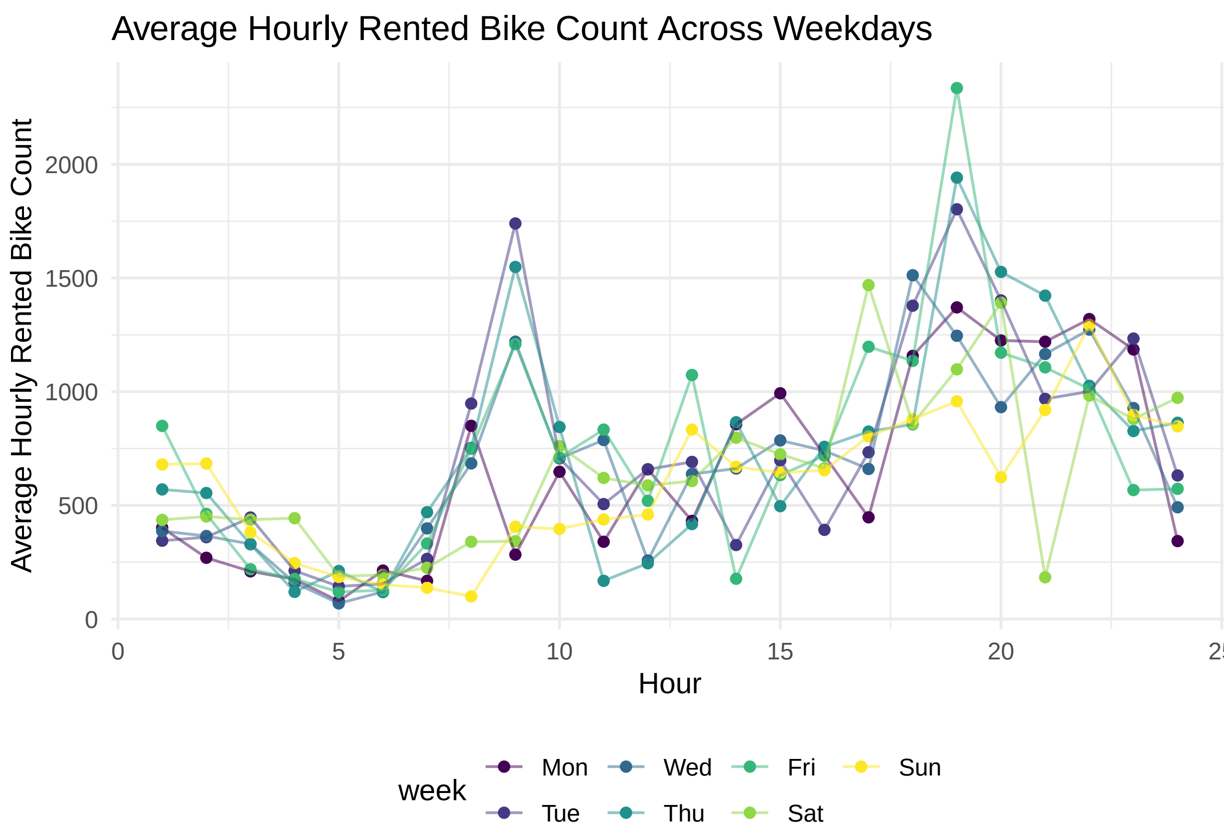
```r
# Summary of Data
# knitr::kable(summary(bike), digits = 2)

# Correlation and Sactter Plot
cont = bike %>% select(-week, -holiday, -func, -season)
ggpairs(cont, lower = list(continuous = wrap("points", alpha = 0.3, size = 0.1)))
```



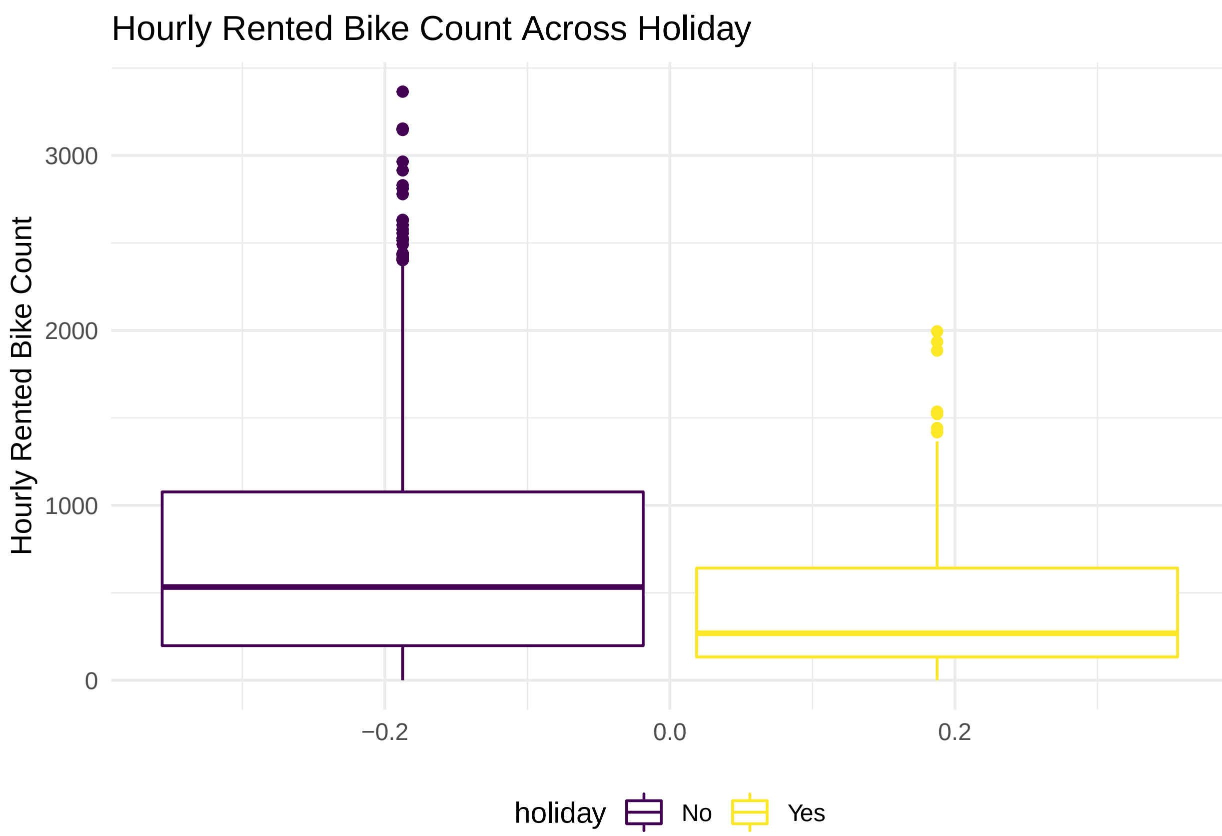

```r
# Average Hourly Rental Bike Count Across Seasons
bike %>%
  mutate(hour = as.factor(hour)) %>%
  group_by(season, hour) %>%
  summarise(rent.avg = mean(rent)) %>%
  mutate(hour = as.integer(hour)) %>%
  ggplot(aes(x = hour, y = rent.avg)) +
  geom_point(aes(color = season)) +
  geom_line(aes(color = season)) +
  labs(
    title = "Average Hourly Rented Bike Count Across Seasons",
    x = "Hour",
    y = "Average Hourly Rented Bike Count")
```



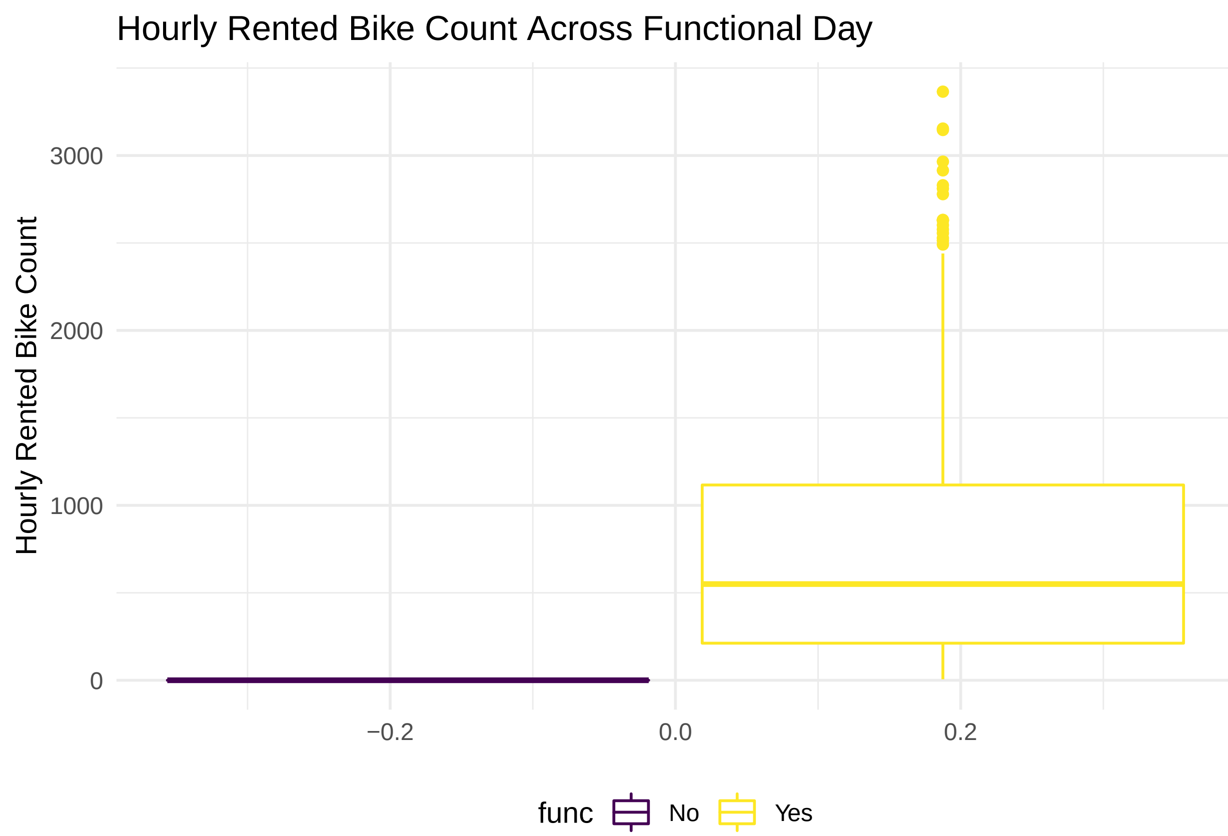

```r
# Average Hourly Rental Bike Count Across Weekdays
bike %>%
  mutate(hour = as.factor(hour)) %>%
  group_by(week, hour) %>%
  summarise(rent.avg = mean(rent)) %>%
  mutate(hour = as.numeric(hour)) %>%
  ggplot(aes(x = hour, y = rent.avg)) +
  geom_point(aes(color = week)) +
  geom_line(aes(color = week), alpha = 0.5) +
  labs(
    title = "Average Hourly Rented Bike Count Across Weekdays",
    x = "Hour",
    y = "Average Hourly Rented Bike Count")
```




```r
# Hourly Rented Bike Count Across Holiday
bike %>%
  ggplot(aes(y = rent, color = holiday)) +
  geom_boxplot() +
  labs(
    title = "Hourly Rented Bike Count Across Holiday",
    y = "Hourly Rented Bike Count")
```




```
# Hourly Rented Bike Count Across Functional Day
bike %>%
  ggplot(aes(y = rent, color = func)) +
  geom_boxplot() +
  labs(
    title = "Hourly Rented Bike Count Across Functional Day",
    y = "Hourly Rented Bike Count")
```



Modeling

LM

```
# Resampling Method - 10-Fold CV
ctrl1 = trainControl(method = "cv", number = 10)

set.seed(2)
lm.fit = train(train_x, train_y,
               method = "lm",
               trControl = ctrl1)
summary(lm.fit)
```

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1120.55  -270.93   -55.25   215.50  1724.37
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -2.461e+02  3.476e+02  -0.708 0.479248
## hour          2.748e+01  2.559e+00  10.739  < 2e-16 ***
```

```
## temp          2.630e+01  1.292e+01   2.036 0.042092 *
## hum          -8.828e+00  3.661e+00  -2.411 0.016121 *
## wind          1.809e+01  1.723e+01   1.050 0.294062
## visibility   -9.858e-03  3.365e-02  -0.293 0.769627
## dew_temp      2.371e+00  1.360e+01   0.174 0.861649
## radiation    -1.220e+02  2.666e+01  -4.575 5.54e-06 ***
## rain         -6.039e+01  9.962e+00  -6.062 2.09e-09 ***
## snow          5.220e+01  4.899e+01   1.066 0.286949
## seasonSpring -2.230e+02  4.792e+01  -4.653 3.84e-06 ***
## seasonSummer -1.768e+02  5.837e+01  -3.029 0.002535 **
## seasonWinter -3.780e+02  6.477e+01  -5.837 7.79e-09 ***
## weekTue       1.112e+02  5.567e+01   1.998 0.046043 *
## weekWed       9.210e+00  5.601e+01   0.164 0.869416
## weekThu       2.041e+02  5.940e+01   3.436 0.000622 ***
## weekFri       1.018e+02  5.751e+01   1.770 0.077156 .
## weekSat      -1.983e+01  5.965e+01  -0.332 0.739611
## weekSun      -1.099e+02  5.898e+01  -1.863 0.062822 .
## holidayYes   -1.089e+02  7.370e+01  -1.478 0.139736
## funcYes       1.057e+03  1.034e+02  10.225  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 436.4 on 779 degrees of freedom
## Multiple R-squared:  0.5747, Adjusted R-squared:  0.5638
## F-statistic: 52.64 on 20 and 779 DF,  p-value: < 2.2e-16
```

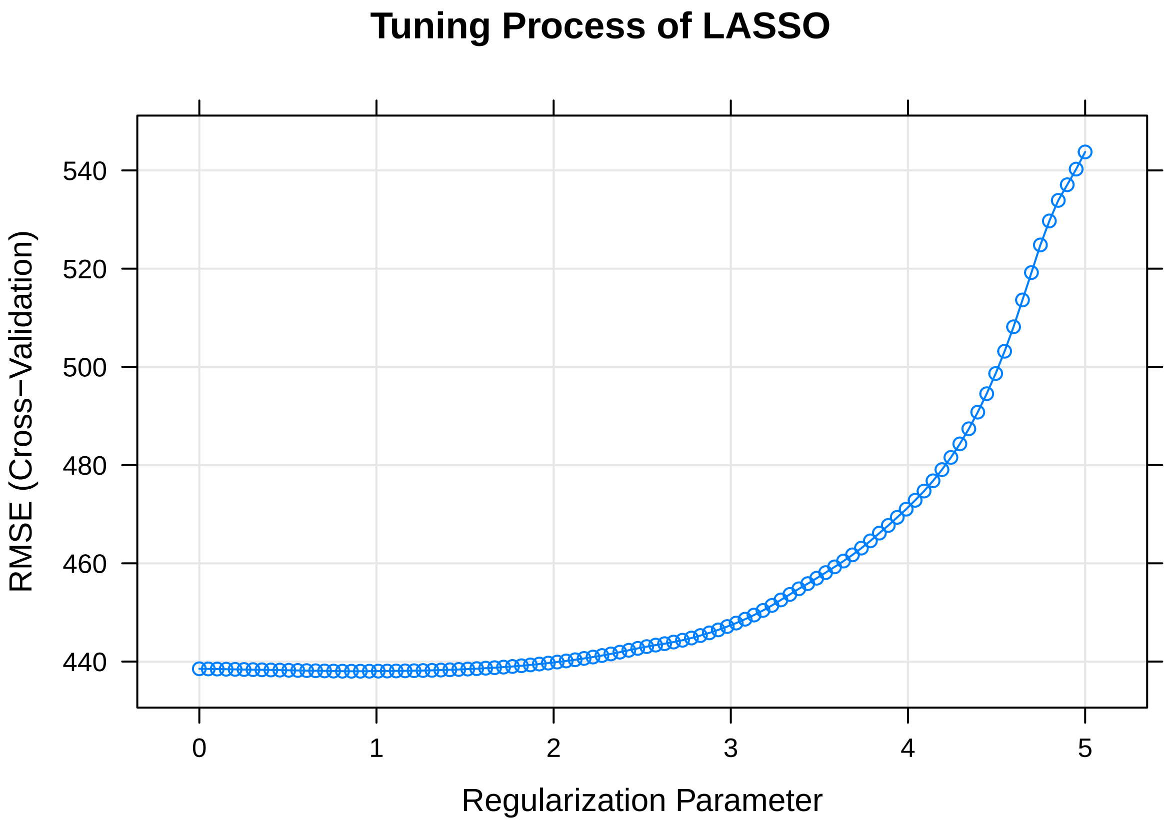
```r
# test error
lm.pred = predict(lm.fit, newdata = test_x)
RMSE(lm.pred, test_y)
```

```
## [1] 396.6971
```

```r
# 396.6971
```

LASSO

```r
set.seed(2)
lasso.fit = train(train_x, train_y,
                  method = "glmnet",
                  tuneGrid = expand.grid(alpha = 1,
                                         lambda = exp(seq(5, 0, length = 100))),
                  trControl = ctrl1)
plot(lasso.fit, xTrans = log, main = "Tuning Process of LASSO")
```




```
lasso.fit$bestTune
```

```
##    alpha   lambda
## 18     1 2.359821
```

```
# lambda = 2.359821

coef(lasso.fit$finalModel, lasso.fit$bestTune$lambda)
```

```
## 21 x 1 sparse Matrix of class "dgCMatrix"
##                       s1
## (Intercept)  -304.00576
## hour           27.71575
## temp           27.33860
## hum            -7.74392
## wind           13.95456
## visibility      .
## dew_temp        .
## radiation    -110.63904
## rain          -59.52354
## snow           38.81008
## seasonSpring -203.01668
## seasonSummer -147.45155
## seasonWinter -367.56817
## weekTue        96.17458
```

```
## weekWed            .
## weekThu     189.14731
## weekFri      86.33245
## weekSat     -24.14187
## weekSun    -113.71925
## holidayYes  -95.42776
## funcYes    1025.85274
```

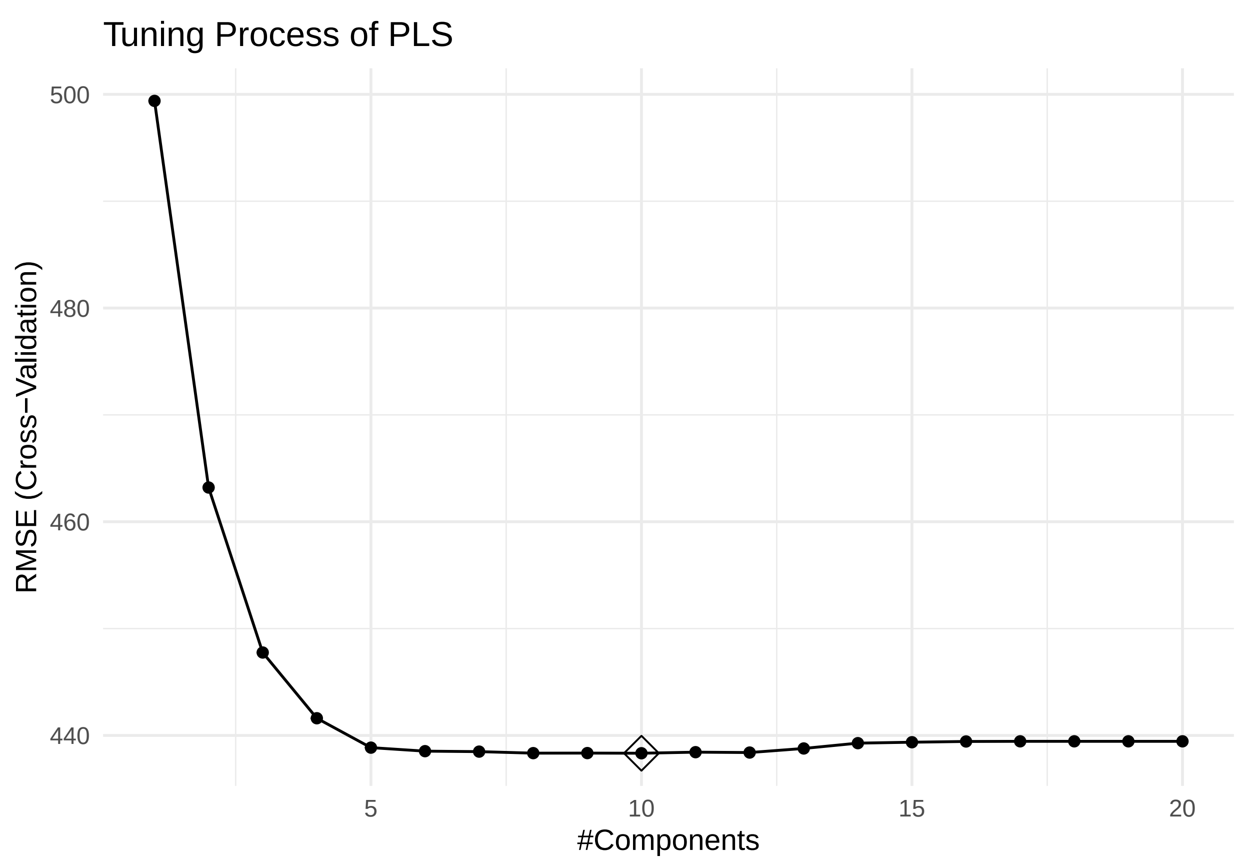
```
# test error
lasso.pred = predict(lasso.fit, newdata = test_x)
RMSE(lasso.pred, test_y)
```

```
## [1] 396.3537
```

```
# 396.3537
```

PLS

```
set.seed(2)
pls.fit = train(train_x, train_y, method = "pls",
                tuneGrid = data.frame(ncomp = 1:20),
                trControl = ctrl1,
                preProcess = c("center", "scale"))
ggplot(pls.fit, highlight = TRUE) +
  labs(title = "Tuning Process of PLS")
```



```r
pls.fit$bestTune
```

```
##    ncomp
## 10    10
```

```r
# ncomp = 10

summary(pls.fit$finalModel)
```

```
## Data:    X dimension: 800 20
##  Y dimension: 800 1
## Fit method: oscorespls
## Number of components considered: 10
## TRAINING: % variance explained
##           1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps
## X           14.34    25.24    32.30    36.07    39.59    43.87    47.86
## .outcome    43.44    51.91    55.62    56.82    57.35    57.42    57.45
##           8 comps  9 comps  10 comps
## X           52.00    56.00     60.14
## .outcome    57.45    57.46     57.46
```

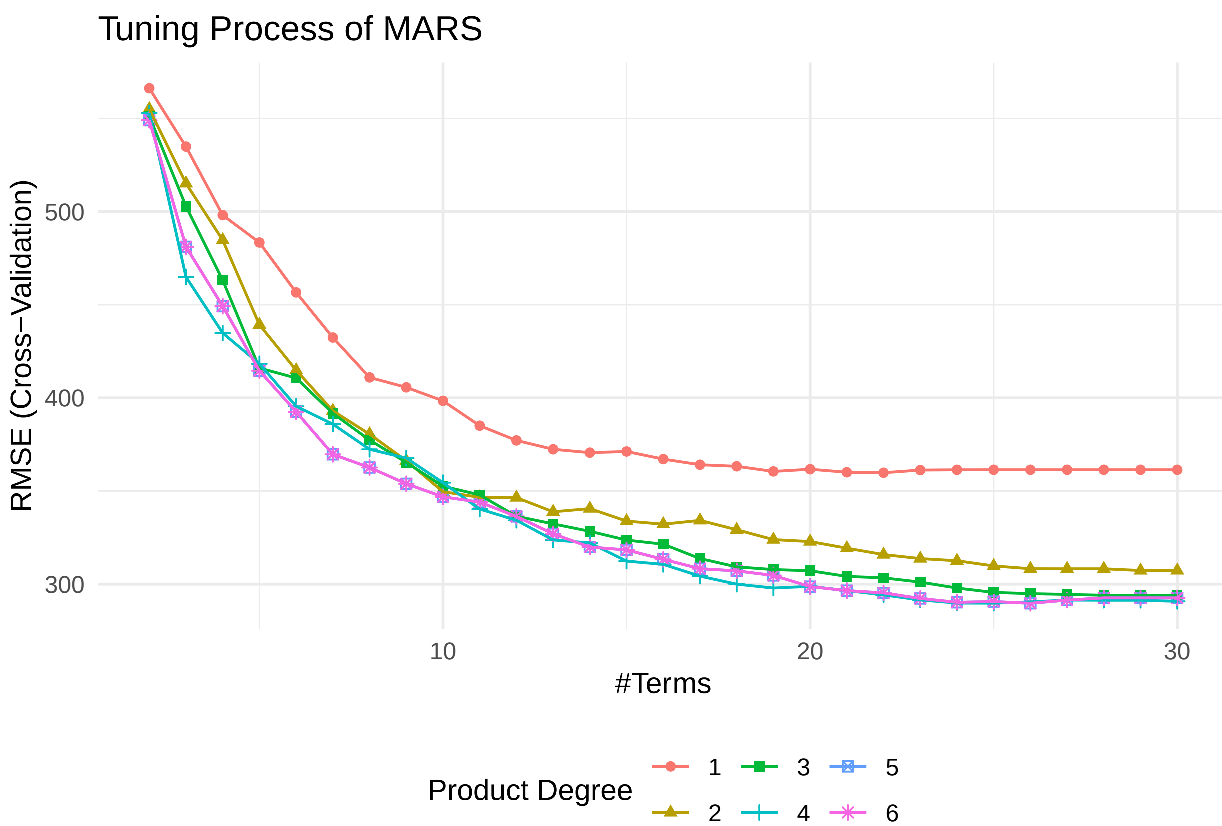
```r
# 57.46% of variance explained

# test error
pls.pred = predict(pls.fit, newdata = test_x, ncomp = ncomp.cv)
RMSE(pls.pred, test_y)
```

```
## [1] 395.9891
```

```r
# 395.9891
```

MARS

```r
set.seed(2)
mars_grid = expand.grid(degree = 1:6, nprune = 2:30)
mars.fit = train(train_x, train_y, method = "earth",
                 tuneGrid = mars_grid,
                 trControl = ctrl1)
ggplot(mars.fit) +
  labs(title = "Tuning Process of MARS")
```




```
mars.fit$bestTune
```

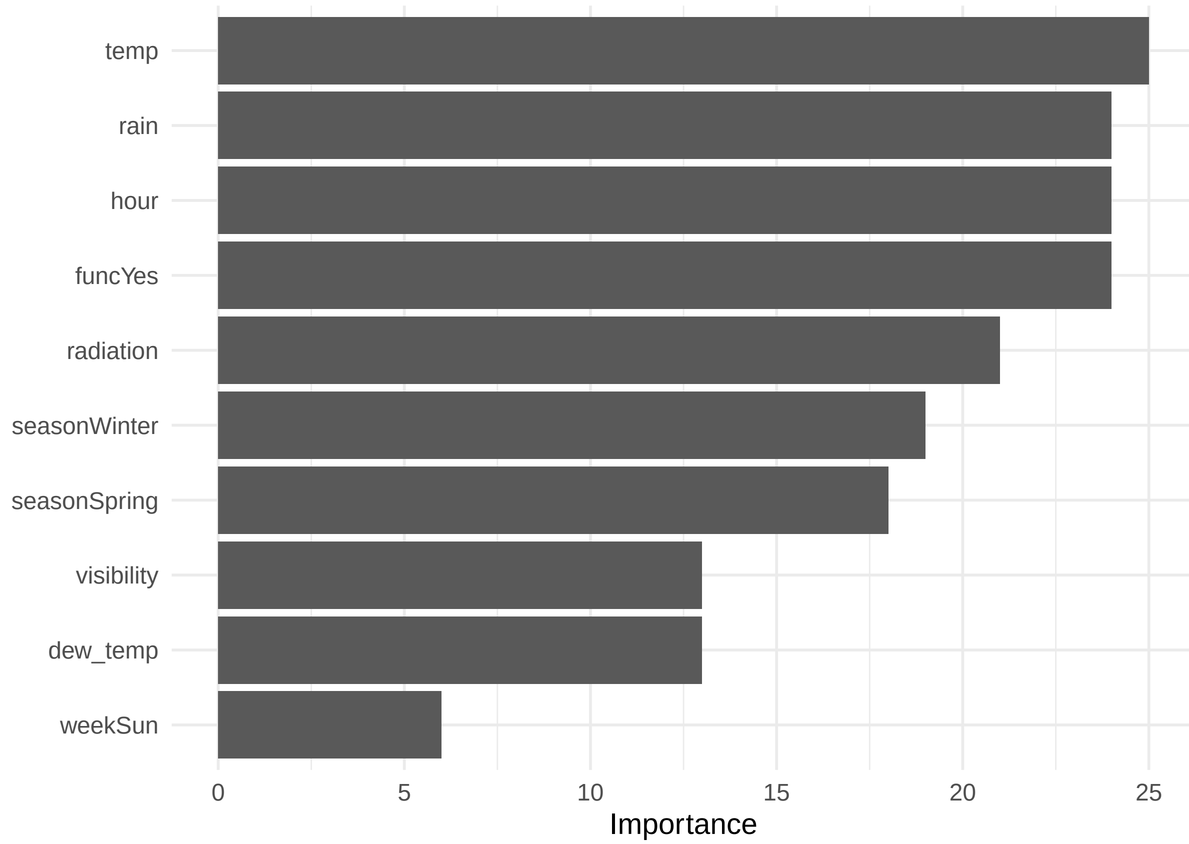
```
##     nprune degree
## 141     26      5
```

```
# nprune = 26; degree = 5

coef(mars.fit$finalModel)
```

```
##                                                     (Intercept)
##                                                   -6.555446e+03
##                                                  h(33.9-temp)
##                                                   -5.868188e+01
##                                                    h(hour-21)
##                                                   -4.923851e+02
##                                                    h(21-hour)
##                                                    5.277675e+01
##                                     h(21-hour) * h(28.8-temp)
##                                                    5.696055e+00
##                                                   h(rain-1.5)
##                                                   -4.671311e+03
##                                                   h(1.5-rain)
##                                                    4.185633e+03
##                                       h(1.5-rain) * funcYes
##                                                    3.210956e+02
```

```
##                                 h(hour-4) * h(1.5-rain) * funcYes
##                                                      6.536085e+01
##                                 h(4-hour) * h(1.5-rain) * funcYes
##                                                      9.545411e+01
##                                          h(8-hour) * h(33.9-temp)
##                                                     -8.846630e+00
##                 h(hour-4) * h(temp-26.9) * h(1.5-rain) * funcYes
##                                                     -6.793200e+00
##                 h(hour-4) * h(26.9-temp) * h(1.5-rain) * funcYes
##                                                     -2.482755e+00
##                       h(hour-8) * h(33.9-temp) * h(9.5-dew_temp)
##                                                      6.892964e-02
##                                                        h(hour-14)
##                                                      1.579968e+02
##                                                 h(radiation-0.03)
##                                                     -6.203958e+03
##                                                 h(0.03-radiation)
##                                                     -9.935974e+03
##                                          h(hour-9) * h(33.9-temp)
##                                                      3.203867e+00
##                                    h(17-hour) * h(radiation-0.03)
##                                                      7.738358e+02
##                                     h(hour-9) * h(radiation-0.03)
##                                                      7.833004e+02
##                                                       h(rain-0.1)
##                                                      4.670069e+03
##                                        h(1.5-rain) * seasonWinter
##                                                     -2.875902e+02
##                                       h(33.9-temp) * seasonSpring
##                                                     -1.176064e+01
##                                   h(1.5-rain) * weekSun * funcYes
##                                                     -1.092494e+02
## h(hour-8) * h(33.9-temp) * h(visibility-460) * h(dew_temp-9.5)
##                                                     -3.339103e-04
## h(hour-8) * h(33.9-temp) * h(460-visibility) * h(dew_temp-9.5)
##                                                     -3.234346e-03
```

```
# Variable Importance Plot
vip(mars.fit$finalModel)
```



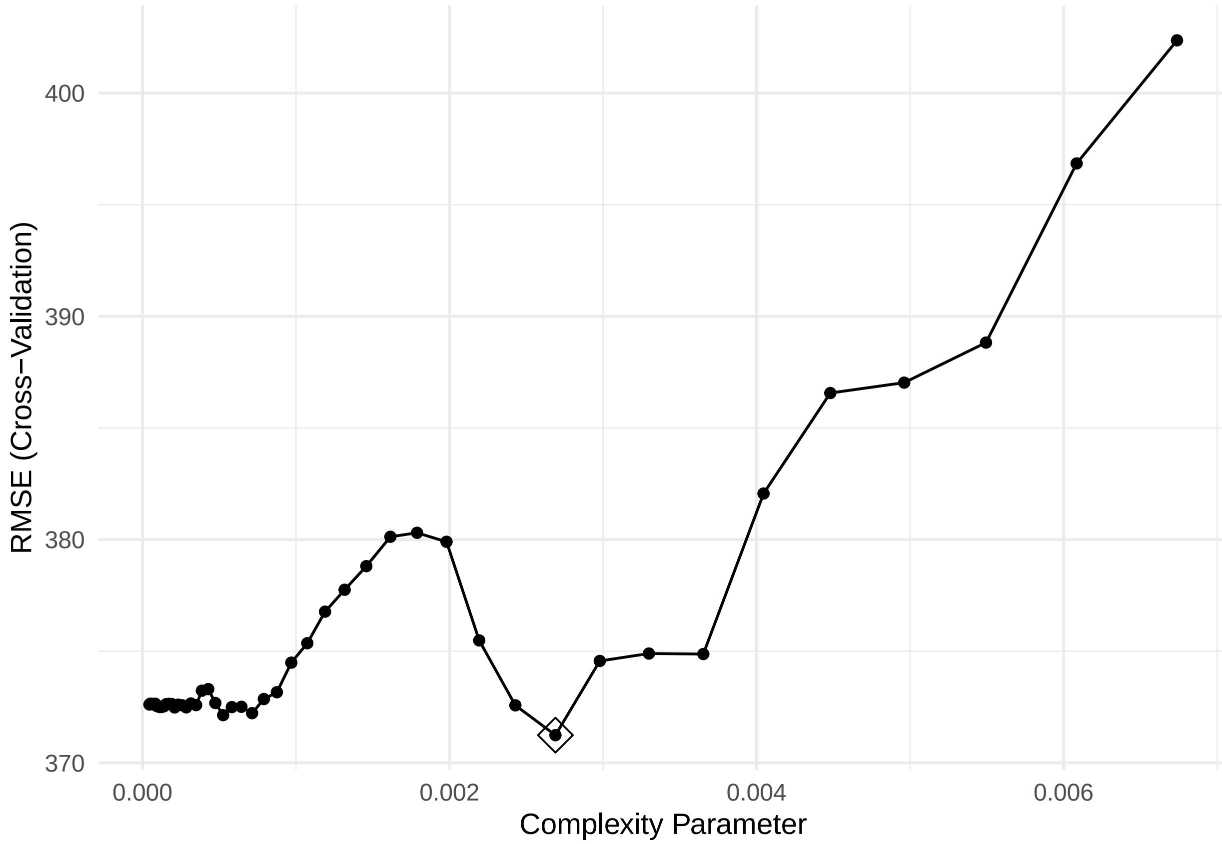

```
# test error
mars.pred = predict(mars.fit, newdata = test_x)
RMSE(mars.pred, test_y)
```

```
## [1] 306.2679
```

```
# 306.2679
```

Regression Tree

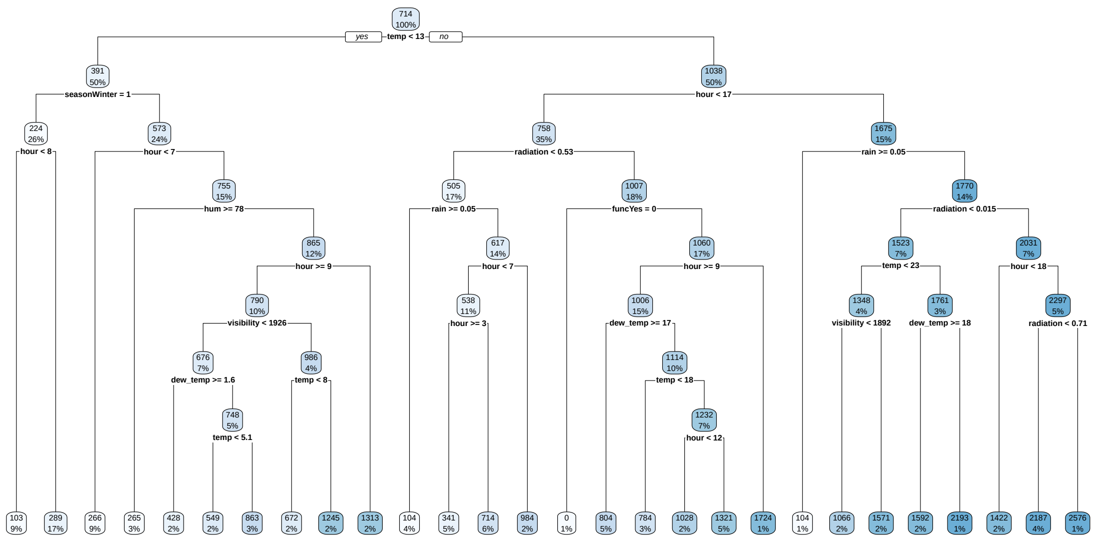
```
set.seed(2)
rpart.fit = train(rent ~.,
                  trainData,
                  method = "rpart",
                  tuneGrid = data.frame(cp = exp(seq(-10, -5, length = 50))),
                  trControl = ctrl1)
ggplot(rpart.fit, highlight = TRUE)
```




```
rpart.fit$bestTune
```

```
##             cp
## 41 0.002689588
```

```
# cp = 0.002689588

# Rpart Plot
rpart.plot(rpart.fit$finalModel)
```



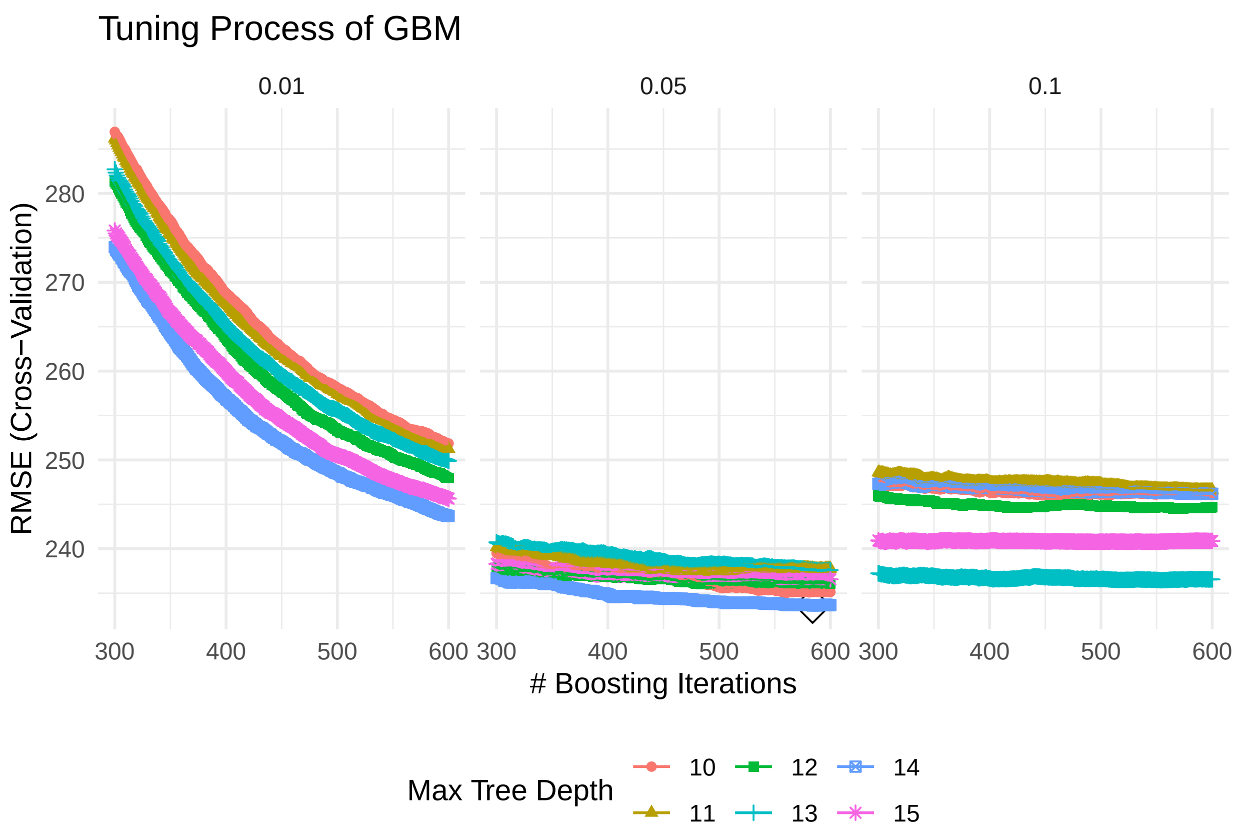

```r
# test error
RMSE(predict(rpart.fit, newdata = testData), test_y)
```

```
## [1] 412.142
```

```r
# 412.142
```

GBM

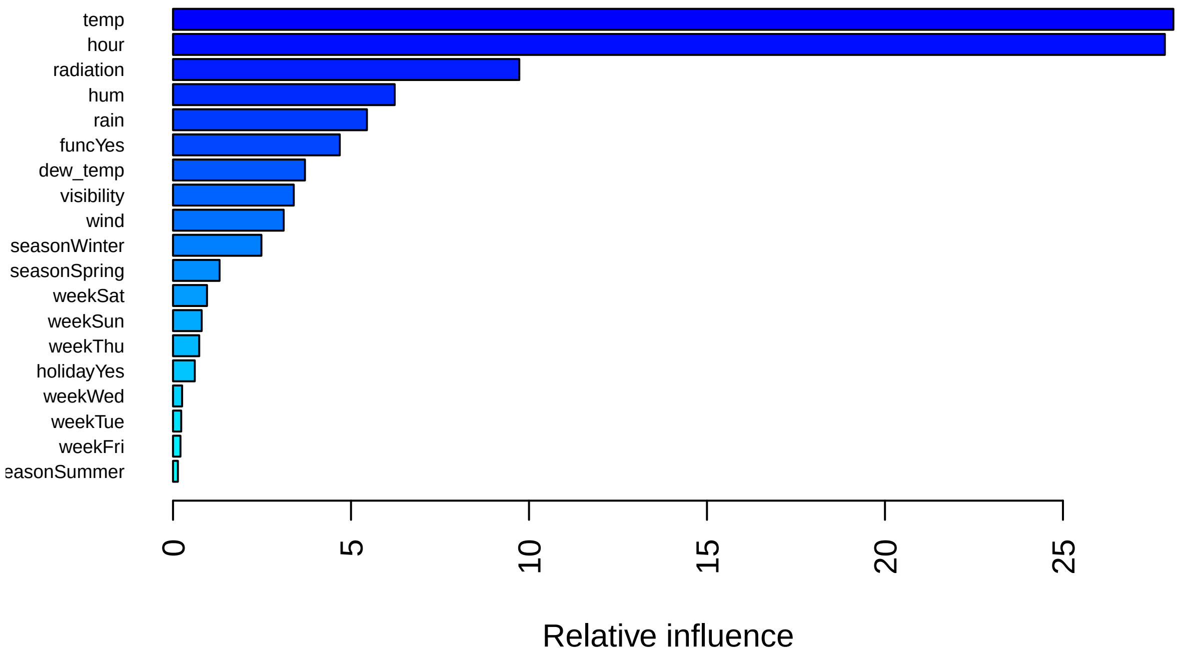
```r
set.seed(2)
gbm.grid = expand.grid(n.trees = 300:600,
                       interaction.depth = 10:15,
                       shrinkage = c(0.01, 0.05, 0.1),
                       n.minobsinnode = 1)
gbm.fit = train(rent ~.,
                trainData,
                method = "gbm",
                tuneGrid = gbm.grid,
                trControl = ctrl1,
                verbose = FALSE)
ggplot(gbm.fit, highlight = TRUE) +
  labs(title = "Tuning Process of GBM")
```




```r
gbm.fit$bestTune
```

```
##      n.trees interaction.depth shrinkage n.minobsinnode
## 3295     584                14      0.05              1
```

```r
# n.trees = 584; interaction.depth = 14; shrinkage = 0.05

# Variable Importance
summary(gbm.fit$finalModel, las = 2, cBars = 19, cex.names = 0.6)
```




```
##                       var     rel.inf
## temp                 temp 28.10098421
## hour                 hour 27.86116128
## radiation       radiation  9.72773812
## hum                   hum  6.22602146
## rain                 rain  5.44645569
## funcYes           funcYes  4.68396078
## dew_temp         dew_temp  3.70596063
## visibility     visibility  3.39116107
## wind                 wind  3.10819802
## seasonWinter seasonWinter  2.48264133
## seasonSpring seasonSpring  1.30705151
## weekSat           weekSat  0.95481430
## weekSun           weekSun  0.80497311
## weekThu           weekThu  0.73593503
## holidayYes     holidayYes  0.61089943
## weekWed           weekWed  0.25528855
## weekTue           weekTue  0.23021666
## weekFri           weekFri  0.20837004
## seasonSummer seasonSummer  0.13614472
## snow                 snow  0.02202402
```

```
# test error
gbm.pred = predict(gbm.fit, newdata = testData)
RMSE(gbm.pred, test_y)
```

```
## [1] 235.5603
```

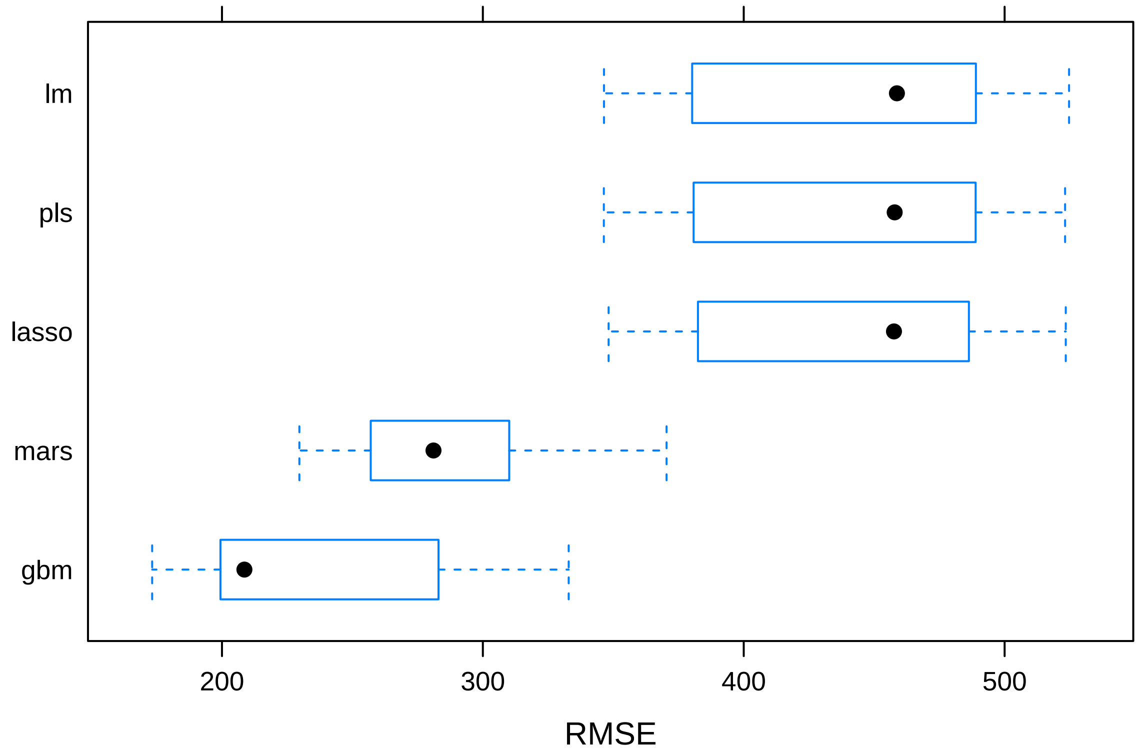
```
# 235.5603
```

Chosing Model - GBM

```
resamp = resamples(list(
  lm = lm.fit,
  lasso = lasso.fit,
  pls = pls.fit,
  mars = mars.fit,
  gbm = gbm.fit))

summary(resamp)
```

```
##
## Call:
## summary.resamples(object = resamp)
##
## Models: lm, lasso, pls, mars, gbm
## Number of resamples: 10
##
## MAE
##           Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
## lm    279.8497 301.7003 345.2637 331.8926 357.1541 373.1838    0
## lasso 282.1144 298.6512 343.1648 330.2047 353.4877 371.5051    0
## pls   279.5372 301.2391 344.9630 331.2658 356.0821 372.9333    0
## mars  174.1918 195.3364 201.9652 201.1253 210.5305 228.2421    0
## gbm   127.7953 138.0474 143.3814 154.2359 173.6313 201.2062    0
##
## RMSE
##           Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
## lm    346.4419 382.0271 458.7146 439.4480 486.4834 524.7071    0
## lasso 348.2079 383.3249 457.5915 438.0258 482.2165 523.4410    0
## pls   346.3796 382.3977 457.8555 438.3322 484.7864 523.1861    0
## mars  229.6639 261.4277 281.0772 289.7088 309.7883 370.4141    0
## gbm   173.2382 199.9303 208.5980 233.6177 268.2053 332.9073    0
##
## Rsquared
##            Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## lm    0.4356089 0.4832034 0.5507070 0.5600693 0.6341620 0.7073765    0
## lasso 0.4387356 0.4897107 0.5538475 0.5624234 0.6327478 0.7050111    0
## pls   0.4388923 0.4881616 0.5522216 0.5620737 0.6341998 0.7074456    0
## mars  0.6974273 0.8068075 0.8230090 0.8089245 0.8480190 0.8707864    0
## gbm   0.7130399 0.8616793 0.9016891 0.8693765 0.9168536 0.9292641    0
```

```
bwplot(resamp, metric = "RMSE")
```

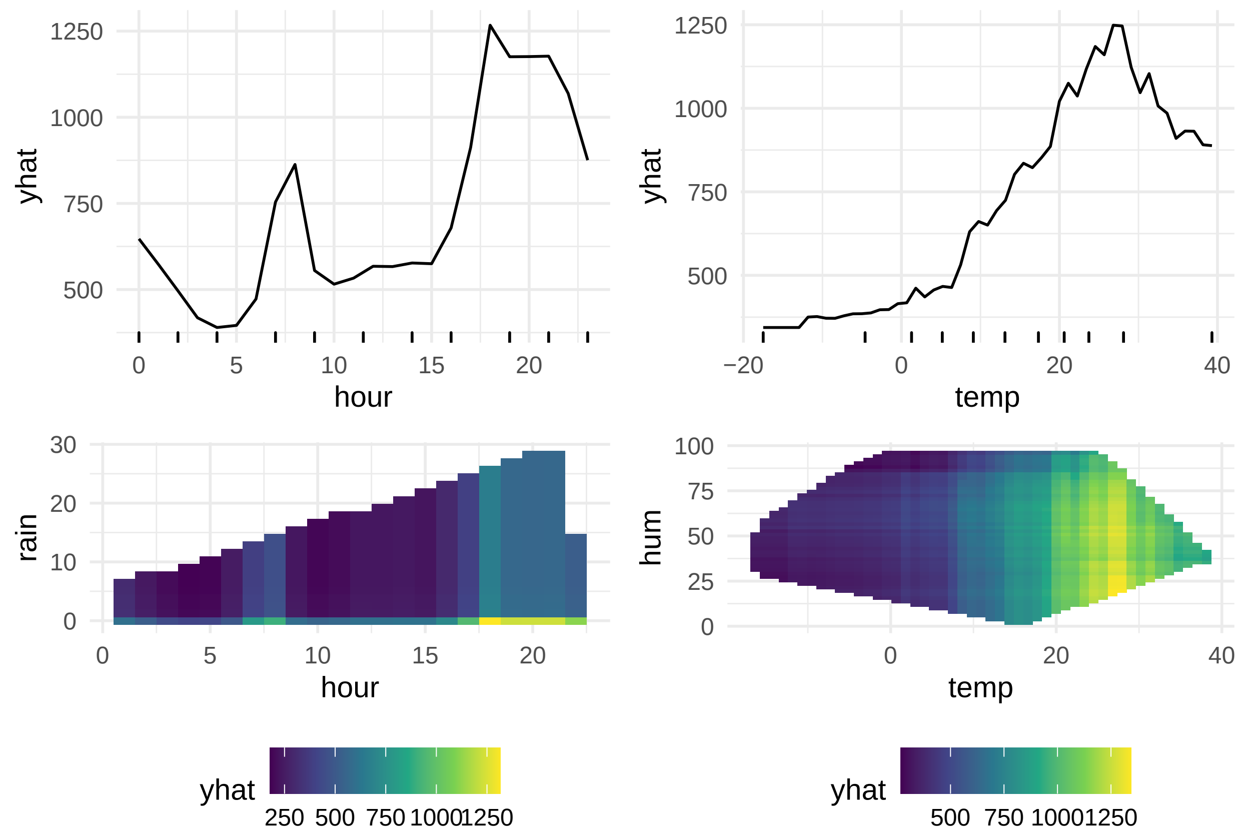


Since the final model is GBM, which is a black-box model…

Black-Box

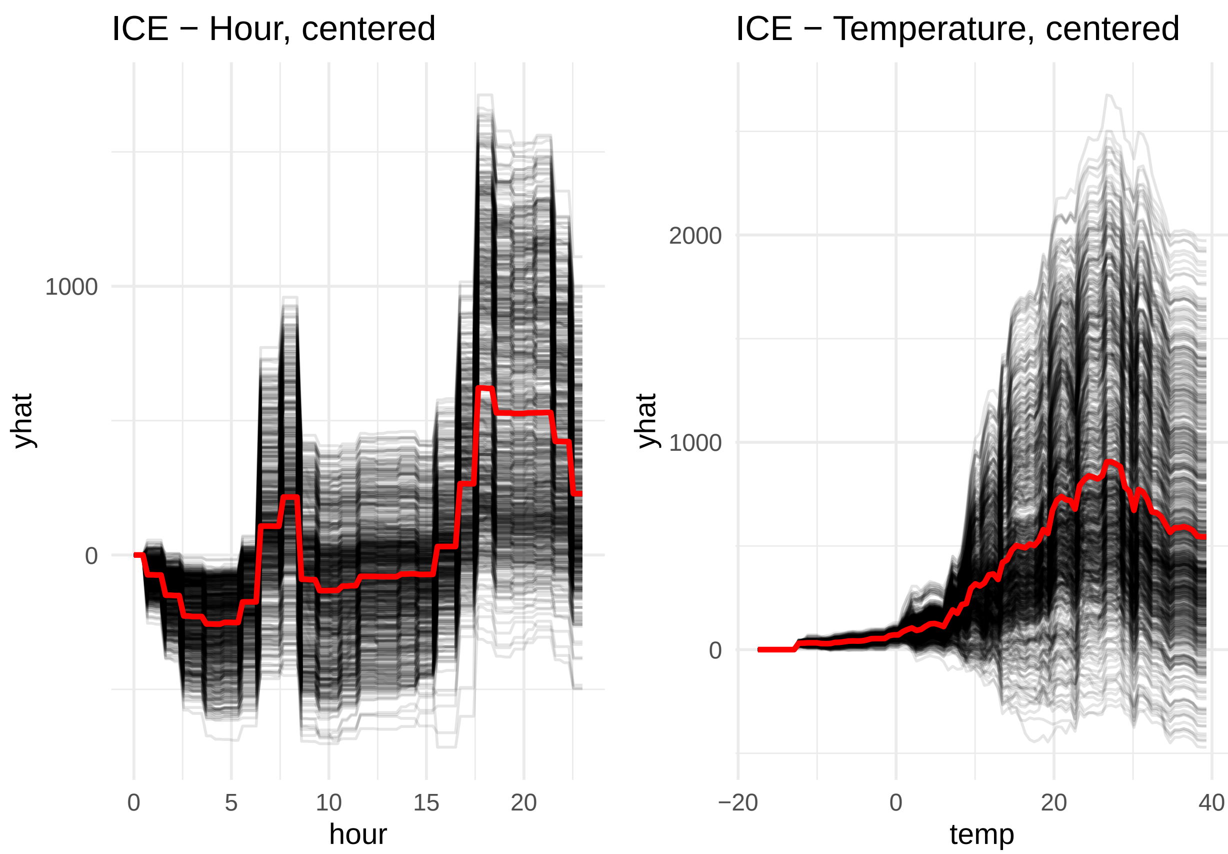
Partial Dependence Plots

```
pdp1.gbm = gbm.fit %>%
  partial(pred.var = c("hour")) %>%
  autoplot(train = trainData, rug = TRUE)

pdp2.gbm = gbm.fit %>%
  partial(pred.var = c("temp")) %>%
  autoplot(train = trainData, rug = TRUE)

pdp3.gbm = gbm.fit %>%
  partial(pred.var = c("hour","rain"), chull = TRUE) %>%
  autoplot(train = trainData, rug = TRUE)

pdp4.gbm = gbm.fit %>%
  partial(pred.var = c("temp","hum"), chull = TRUE) %>%
  autoplot(train = trainData, rug = TRUE)

grid.arrange(pdp1.gbm, pdp2.gbm, pdp3.gbm, pdp4.gbm, ncol = 2, nrow = 2)
```



ICE curves

```
ice1.gbm <- gbm.fit %>%
  partial(pred.var = "hour",
          grid.resolution = 100,
          ice = TRUE) %>%
  autoplot(train = bike, alpha = .1,
           center = TRUE) +
  ggtitle("ICE - Hour, centered")

ice2.gbm <- gbm.fit %>%
  partial(pred.var = "temp",
          grid.resolution = 100,
          ice = TRUE) %>%
  autoplot(train = bike, alpha = .1,
           center = TRUE) +
  ggtitle("ICE - Temperature, centered")

grid.arrange(ice1.gbm, ice2.gbm, nrow = 1)
```


ICE − Hour, centered
yhat
1000
0
0
5
10
15
20
hour
ICE − Temperature, centered
yhat
2000
1000
0
−20
0
20
40
temp