HedonicModel2:SND

luismor

3/26/2021

VARIABLES SELECTION

Loading the data

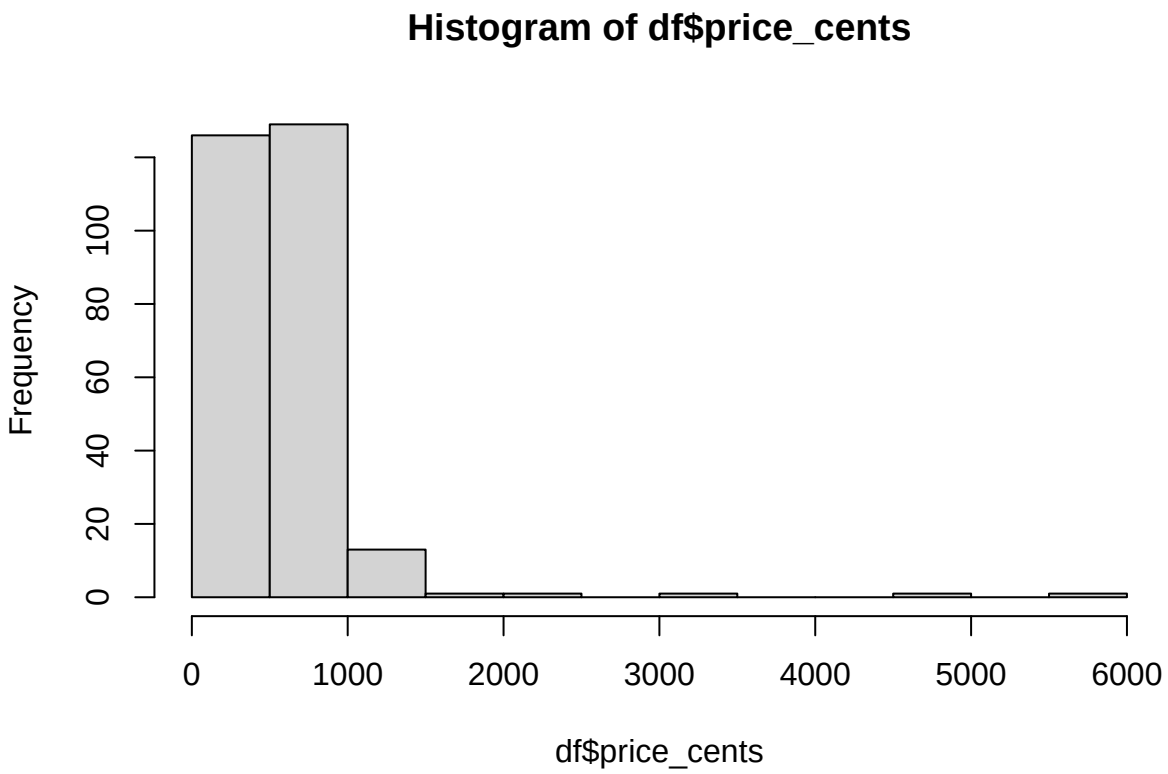
```
library(readxl)
df <- read_excel("/Users/Unimooc/Dropbox/2021/Directorio R/SpaceNextDoor/NextDoor/Optimal pricing/NextD
                 sheet = "Data1")
```

```
head(df)
```

Visualization

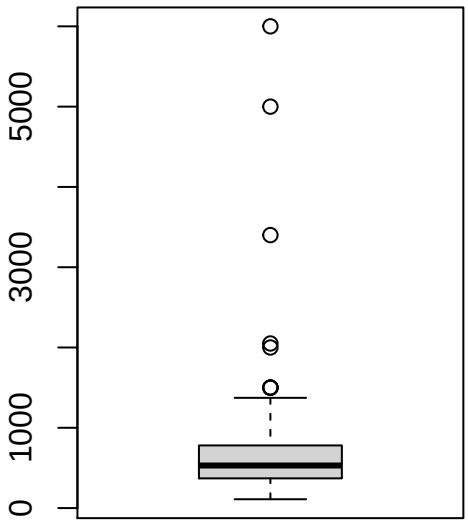
```
library(ggplot2)
library(corrplot)
library(tidyverse)
library(MASS)
```

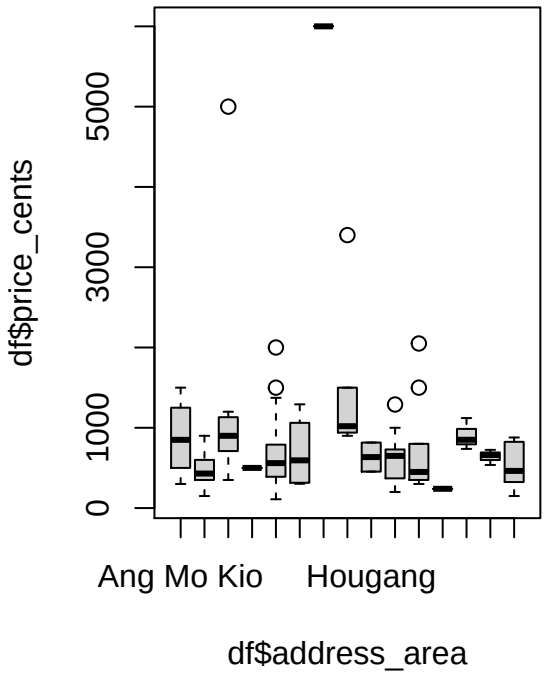
```
hist(df$price_cents)
```




```
#The histogram shows a positive skweness (positioned to the left). There are many outliers that should
```

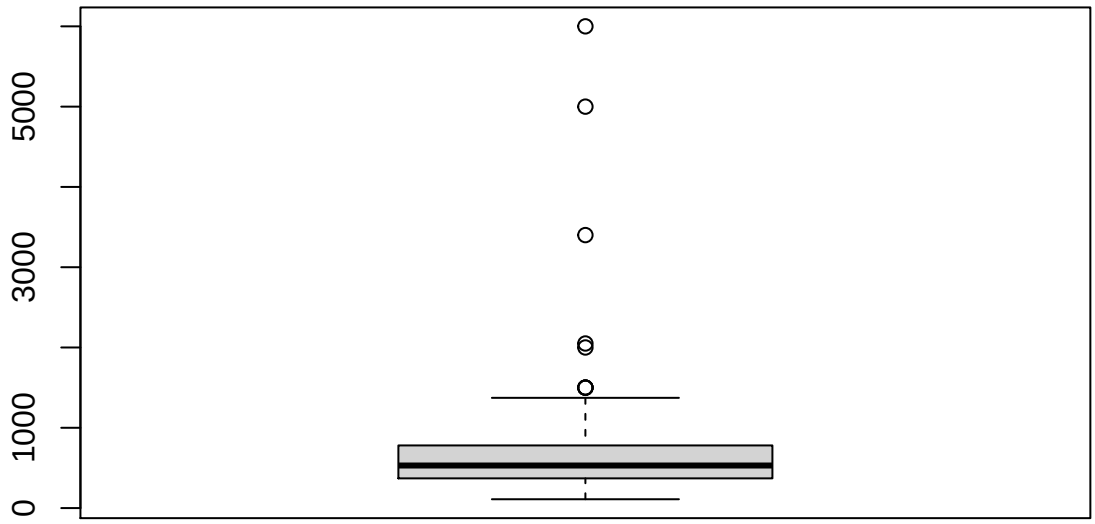
```
par(mfrow=c(1,2))

  boxplot(df$price_cents)
  boxplot(df$price_cents ~ df$address_area)
```



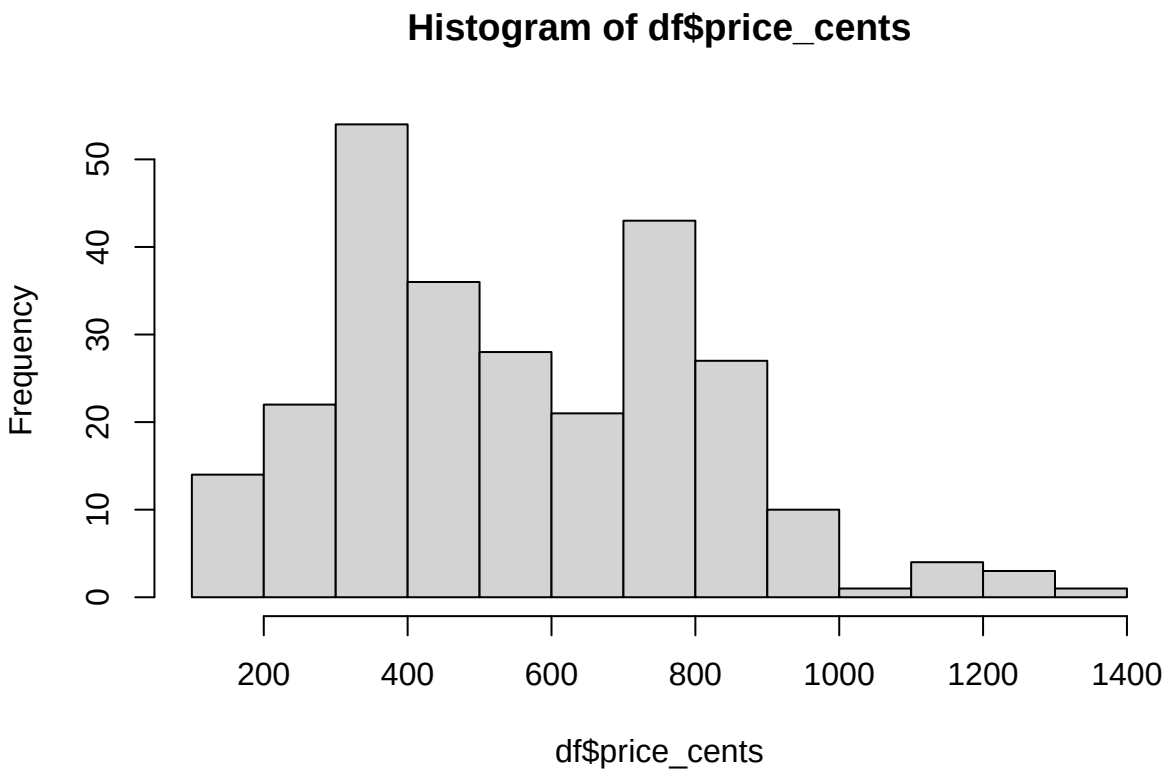



```
#The Boxplot confirms the presence of outliers caused by some isolated data in a few districts. As we g
```

```
gcaja <- boxplot(df$price_cents)
```

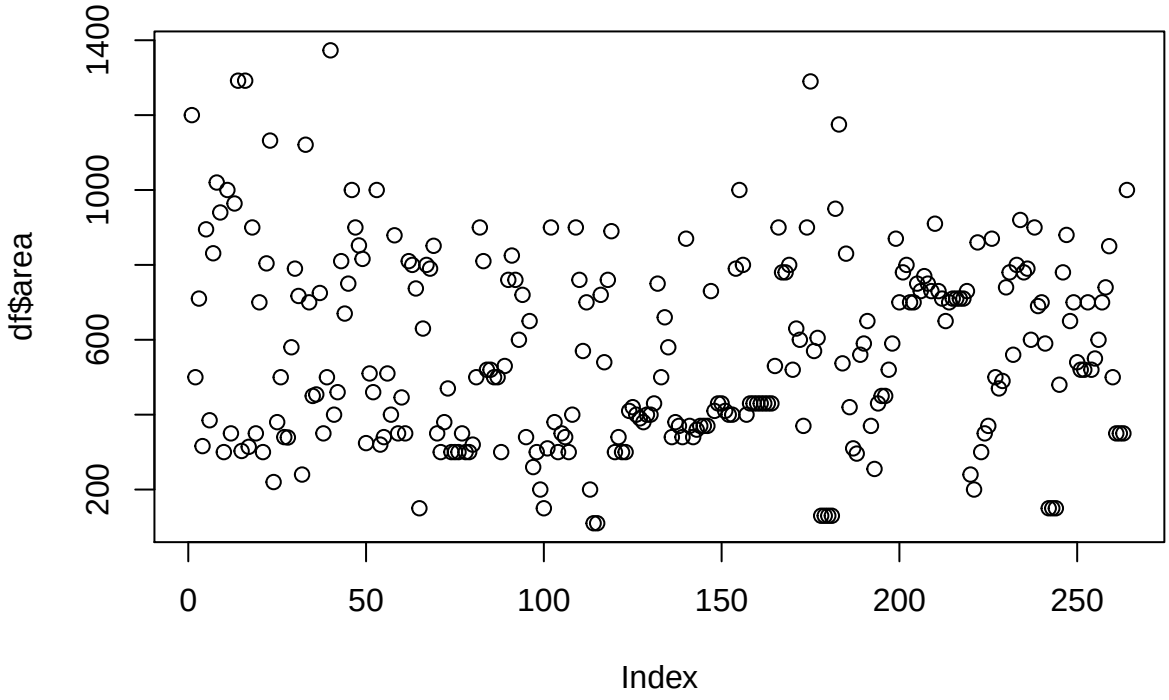



```
df<-df[!(df$price_cents %in% gcaja$out),]
hist(df$price_cents)
```



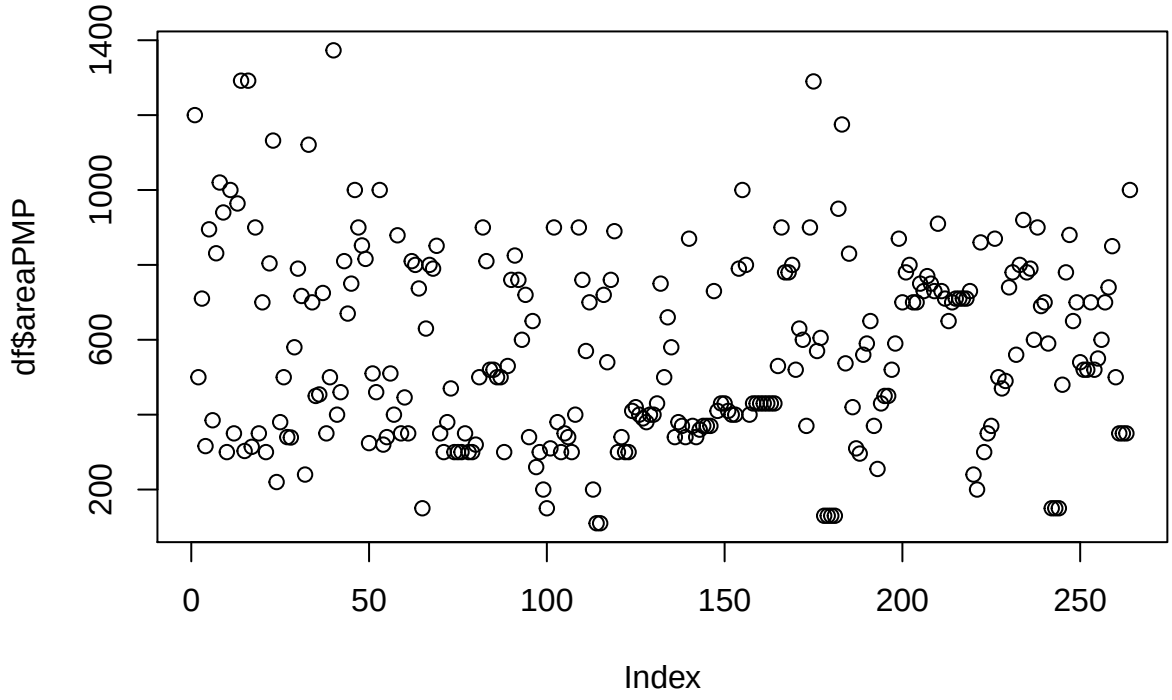

```
plot(df$price_cents,df$area)
```

```
## Warning: Unknown or uninitialised column: 'area'.
```




```
plot(df$price_cents,df$areaPMP)
```

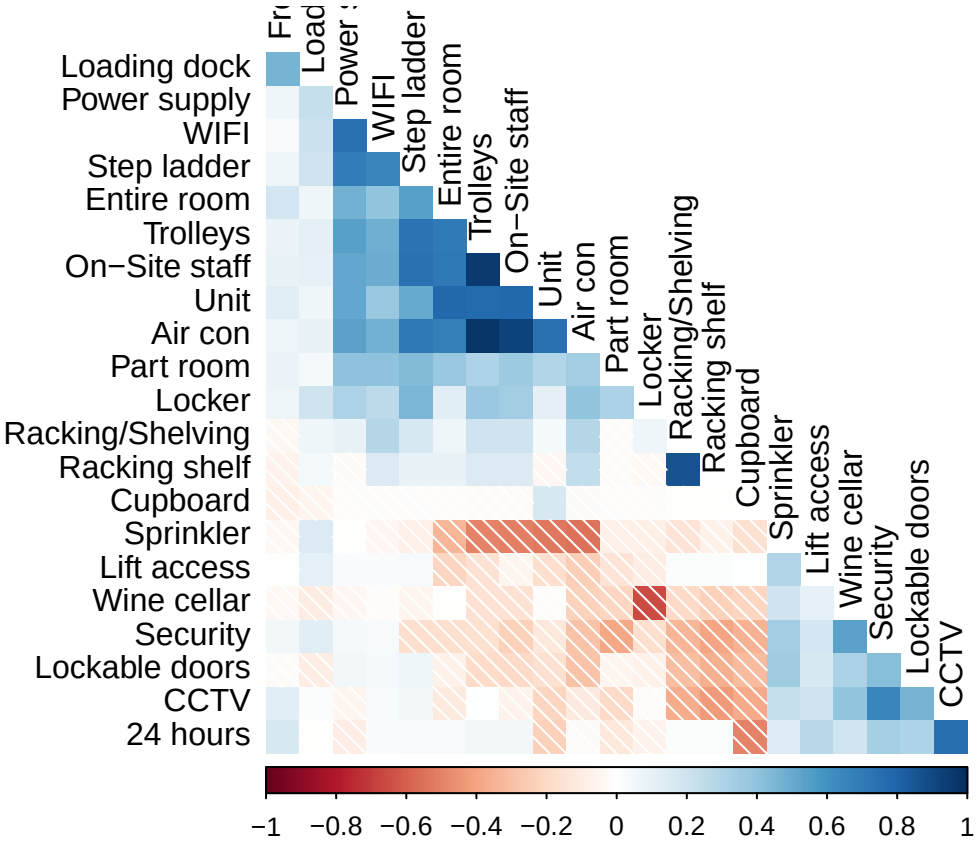
```
## Warning: Unknown or uninitialised column: ‘areaPMP‘.
```




```
#Deleting outlyers
```

```
df.cor <- cor(df[,c(21:42)], method = "kendall")
round(df.cor, digits = 1)
```

```
corrplot(df.cor, method ="shade",
         tl.col ="black",
         order = "AOE", type = "lower", diag = F)
```




```
#The correlation matrix indicates the presence of strong autocorrelation between some variables. We sho
```

Training and test sample division

```
library(dplyr)
library(caret)
```

```
df.sel <- df[,-c(1,3,4,6:20)]

set.seed(2021)
dfPartition <- createDataPartition(y = df.sel$price_cents,
                                   p = 0.7, list = F)

Training <- df.sel[dfPartition,]
```

```
## Warning: The 'i' argument of ''['()' can't be a matrix as of tibble 3.0.0.
## Convert to a vector.
## This warning is displayed once every 8 hours.
## Call 'lifecycle::last_warnings()' to see where this warning was generated.
```

```
Test <- df.sel[-dfPartition,]
```

Variables selection

AIC forward selection

```
Modelzero <- lm(price_cents~1,data=Training)
  summary(Modelzero)

  FitAll = lm(price_cents ~ ., data=Training)
  formula(FitAll)

  model.forward <- step(Modelzero,direction="forward",scope=formula(FitAll))
```

```
summary(model.forward)
```

```
##
## Call:
## lm(formula = price_cents ~ `area squared foot` + Unit + `Racking shelf` +
##     `Wine cellar` + `Lift access` + `Loading dock` + `Free Parking` +
##     `Step ladder` + Locker + `Part room` + `Power supply` + `24 hours` +
##     CCTV + Security, data = Training)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -422.14  -52.80    0.49   63.12  409.28
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)         1029.817    123.166   8.361 1.98e-14 ***
## `area squared foot`   56.991      2.426  23.488  < 2e-16 ***
## Unit                 197.578     34.710   5.692 5.27e-08 ***
## `Racking shelf`      367.744    120.546   3.051 0.002643 **
## `Wine cellar`         -4.303     72.533  -0.059 0.952762
## `Lift access`       -367.256     92.629  -3.965 0.000107 ***
## `Loading dock`        95.702     20.595   4.647 6.65e-06 ***
## `Free Parking`       -60.695     20.704  -2.932 0.003829 **
## `Step ladder`        207.171     57.519   3.602 0.000413 ***
## Locker              -200.357     62.580  -3.202 0.001627 **
## `Part room`         -291.327     72.627  -4.011 8.97e-05 ***
## `Power supply`       -57.960     54.931  -1.055 0.292824
## `24 hours`          -754.565    206.406  -3.656 0.000340 ***
## CCTV                 693.218    212.055   3.269 0.001302 **
## Security            -349.596    140.300  -2.492 0.013652 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 104.8 on 173 degrees of freedom
## Multiple R-squared:  0.8362, Adjusted R-squared:  0.823
## F-statistic: 63.09 on 14 and 173 DF,  p-value: < 2.2e-16
```

```
predict.for.tr <- predict(model.forward, newdata = Training)
```

```r
training.for.mse <- mean((predict.for.tr - Training$price_cents)^2)
paste("Training MSE error:", training.for.mse)
```

```
## [1] "Training MSE error: 10106.646151509"
```

```r
predict.for.tst <- predict(model.forward, newdata = Test)

test.for.mse <- mean((predict.for.tst - Test$price_cents)^2)
paste("Test MSE error:", test.for.mse)
```

```
## [1] "Test MSE error: 27134.847465538"
```

AIC backward selection

```r
model.backward <- stepAIC(FitAll, trace=TRUE, direction="backward")
```

```r
summary(model.backward)
```

```
##
## Call:
## lm(formula = price_cents ~ `area squared foot` + Locker + `Racking shelf` +
##     `Part room` + Unit + `24 hours` + `Lift access` + Security +
##     `Loading dock` + CCTV + `On-Site staff` + `Free Parking` +
##     Trolleys + `Step ladder` + `Power supply`, data = Training)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -421.36  -51.79    0.00   62.14  424.37
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)         1023.563    116.632   8.776 1.63e-15 ***
## `area squared foot`   56.905      2.397  23.743  < 2e-16 ***
## Locker              -217.333     36.478  -5.958 1.41e-08 ***
## `Racking shelf`      331.911    120.496   2.755 0.006510 **
## `Part room`         -270.670     73.349  -3.690 0.000300 ***
## Unit                 192.372     61.051   3.151 0.001919 **
## `24 hours`          -719.080    205.760  -3.495 0.000603 ***
## `Lift access`       -365.275     93.235  -3.918 0.000129 ***
## Security            -314.812    130.608  -2.410 0.016991 *
## `Loading dock`        93.374     20.285   4.603 8.06e-06 ***
## CCTV                 621.127    219.930   2.824 0.005300 **
## `On-Site staff`     -235.203    139.639  -1.684 0.093926 .
## `Free Parking`       -55.645     20.831  -2.671 0.008282 **
## Trolleys             242.517    125.542   1.932 0.055033 .
## `Step ladder`        239.934     63.204   3.796 0.000203 ***
## `Power supply`      -108.467     58.492  -1.854 0.065397 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 104 on 172 degrees of freedom
## Multiple R-squared:  0.8397, Adjusted R-squared:  0.8257
## F-statistic: 60.06 on 15 and 172 DF,  p-value: < 2.2e-16
```

```
predict.bck.tr <- predict(model.backward, newdata = Training)

training.bck.mse <- mean((predict.bck.tr - Training$price_cents)^2)
paste("Training MSE error:", training.bck.mse)
```

Summary

```
## [1] "Training MSE error: 9892.19045589417"
```

```
predict.bck.tst <- predict(model.backward, newdata = Test)

test.bck.mse <- mean((predict.bck.tst - Test$price_cents)^2)
paste("Test MSE error:", test.bck.mse)
```

```
## [1] "Test MSE error: 31053.1464613034"
```

Ridge and Lasso regularizations

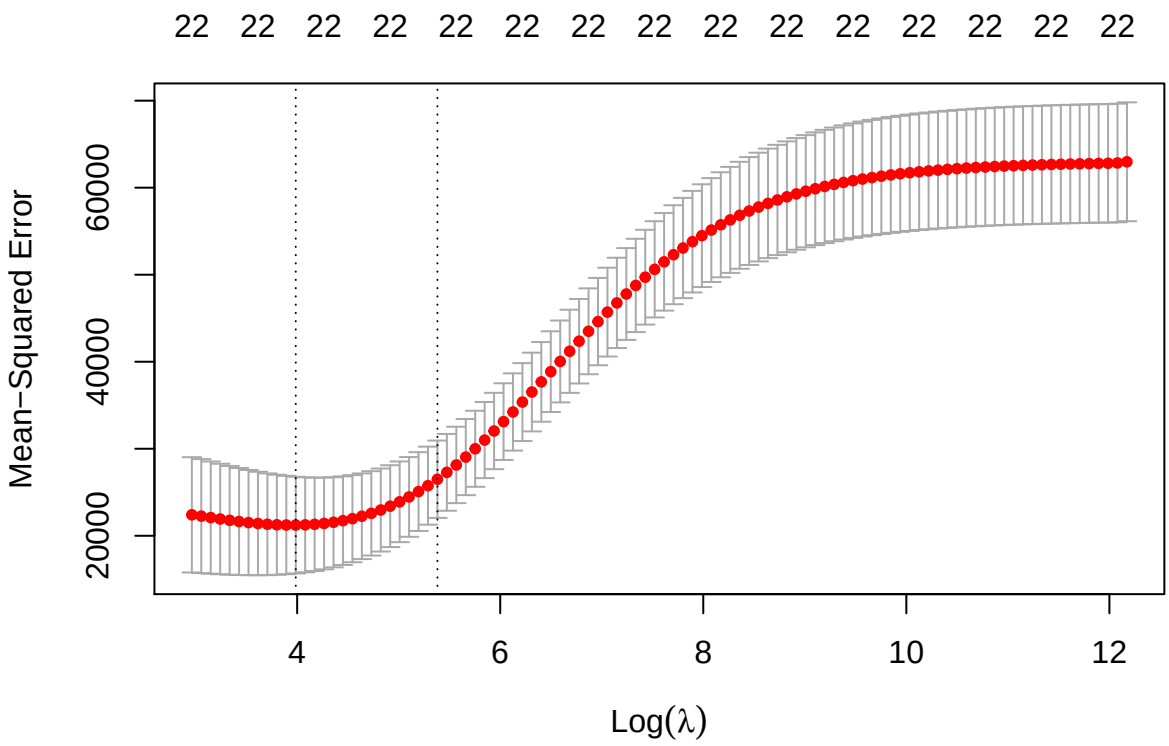
```
library(glmnet)
```

```
## Loading required package: Matrix

##
## Attaching package: 'Matrix'

## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack

## Loaded glmnet 4.0-2
```

```
# Convert into a matrix train and test data
train.mat <- model.matrix(price_cents ~ ., data = Training)
test.mat <- model.matrix(price_cents ~ ., data = Test)
```

Ridge

```
# Cross validation to obtain the best value of lambda. Error evolution.
cv.ridge <- cv.glmnet(x = train.mat, y = Training$price_cents, alpha = 0,
                      lambda = NULL, type.measure="mse")

plot(cv.ridge)
```




```
paste("Best lambda:", cv.ridge$lambda.min)
```

```
## [1] "Best lambda: 53.8407349869104"
```

```
paste("Best lambda + y sd:", cv.ridge$lambda.1se)
```

```
## [1] "Best lambda + y sd: 217.35597607519"
```

```
# Training the model
mod.ridge.train <- glmnet(x = train.mat, y = Training$price_cents, alpha = 0,
                          lambda = cv.ridge$lambda.1se)

dim(coef(mod.ridge.train))
```

```
## [1] 25  1
```

```
coef(mod.ridge.train, s = "lambda.1se")
```

```
## 25 x 1 sparse Matrix of class "dgCMatrix"
##                                1
## (Intercept)          650.0109546
## (Intercept)            .
## `area squared foot`   28.4014079
```

```
## Locker              -102.6060579
## Cupboard               .
## `Racking shelf`       74.6947435
## `Part room`          -62.0198794
## `Entire room`         41.9460348
## Unit                  52.1683007
## `Wine cellar`         99.4284252
## `Air con`             23.5066522
## `24 hours`          -133.1727289
## `Lift access`       -177.0092989
## Security            -13.9526441
## `Loading dock`        32.0520610
## CCTV                -53.6492879
## `Lockable doors`      22.2000478
## `On-Site staff`       26.4659750
## `Free Parking`         5.2805235
## Trolleys              25.9078575
## `Step ladder`         33.6781259
## `Racking/Shelving`    74.5268171
## Sprinkler             -0.9940603
## `Power supply`        -1.7195305
## WIFI                -13.4527242
```

```
# Training predictions
pred.ridge <- predict(mod.ridge.train, newx = train.mat)

# Training error (MSE)
tr.ridge.mse <- mean((pred.ridge - Training$price_cents)^2)
paste("Training MSE error:", tr.ridge.mse)
```

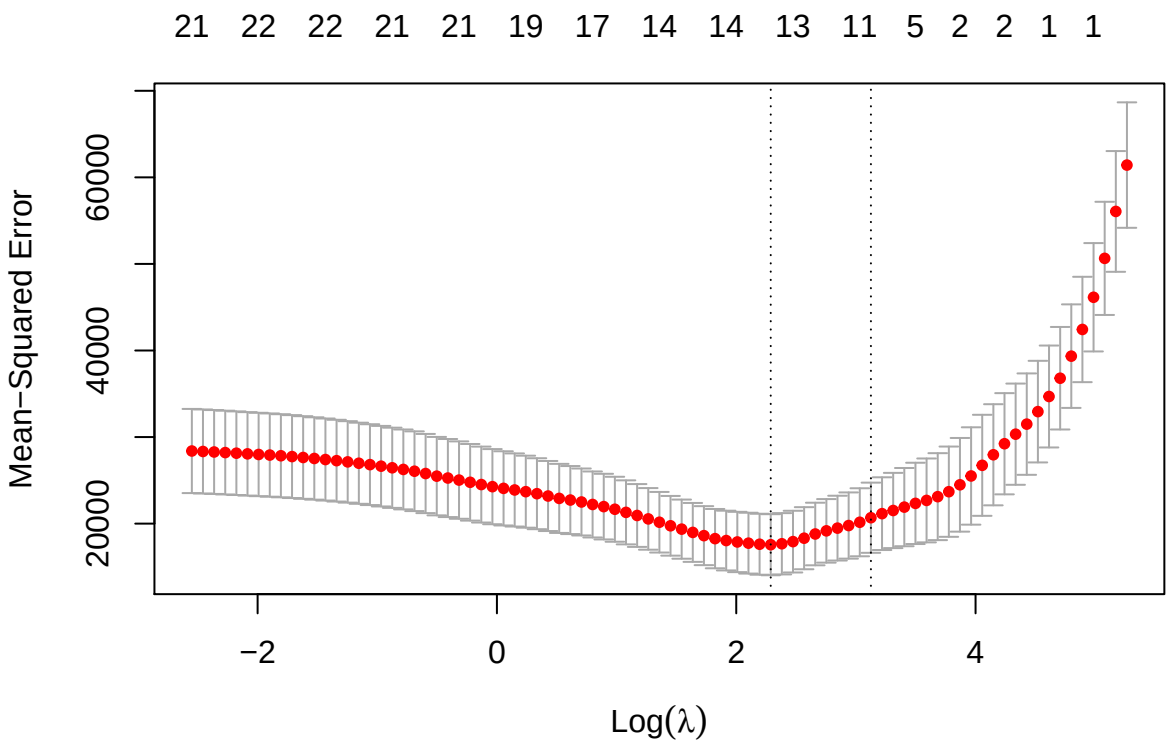
```
## [1] "Training MSE error: 21264.3271640064"
```

```
#Test predictions: using training model
pred.test.ridge <- predict(mod.ridge.train,newx = test.mat)

test.ridge.mse <- mean((pred.test.ridge - Test$price_cents)^2)
paste("Test MSE error:",test.ridge.mse)
```

```
## [1] "Test MSE error: 32160.8355527727"
```

Lasso

```
cv.lasso <- cv.glmnet(x = train.mat, y = Training$price_cents, alpha = 1,
                      lambda = NULL, type.measure="mse")

plot(cv.lasso)
```




```
paste("Best lambda:", cv.lasso$lambda.min)
```

```
## [1] "Best lambda: 9.85682941015129"
```

```
paste("Best lambda + y sd:", cv.lasso$lambda.1se)
```

```
## [1] "Best lambda + y sd: 22.7705543512872"
```

```
# Training the model
mod.lasso.train <- glmnet(x = train.mat, y = Training$price_cents, alpha = 1,
                          lambda = cv.lasso$lambda.1se)

dim(coef(mod.lasso.train))
```

```
## [1] 25  1
```

```
coef(mod.lasso.train, s = "lambda.1se")
```

```
## 25 x 1 sparse Matrix of class "dgCMatrix"
##                                1
## (Intercept)         424.523884
## (Intercept)           .
## `area squared foot`  49.208603
```

```
## Locker              -88.719618
## Cupboard              .
## `Racking shelf`       .
## `Part room`           .
## `Entire room`         .
## Unit                105.572855
## `Wine cellar`         .
## `Air con`            48.209032
## `24 hours`          -37.990970
## `Lift access`       -91.682152
## Security              .
## `Loading dock`       10.750152
## CCTV                  .
## `Lockable doors`      .
## `On-Site staff`       .
## `Free Parking`        .
## Trolleys              .
## `Step ladder`        40.613963
## `Racking/Shelving`    3.215173
## Sprinkler             .
## `Power supply`        .
## WIFI                  .
```

```
# Training predictions
pred.lasso <- predict(mod.lasso.train, newx = train.mat)

# Training error (MSE)
tr.lasso.mse <- mean((pred.lasso - Training$price_cents)^2)
paste("Training MSE error:", tr.lasso.mse)
```

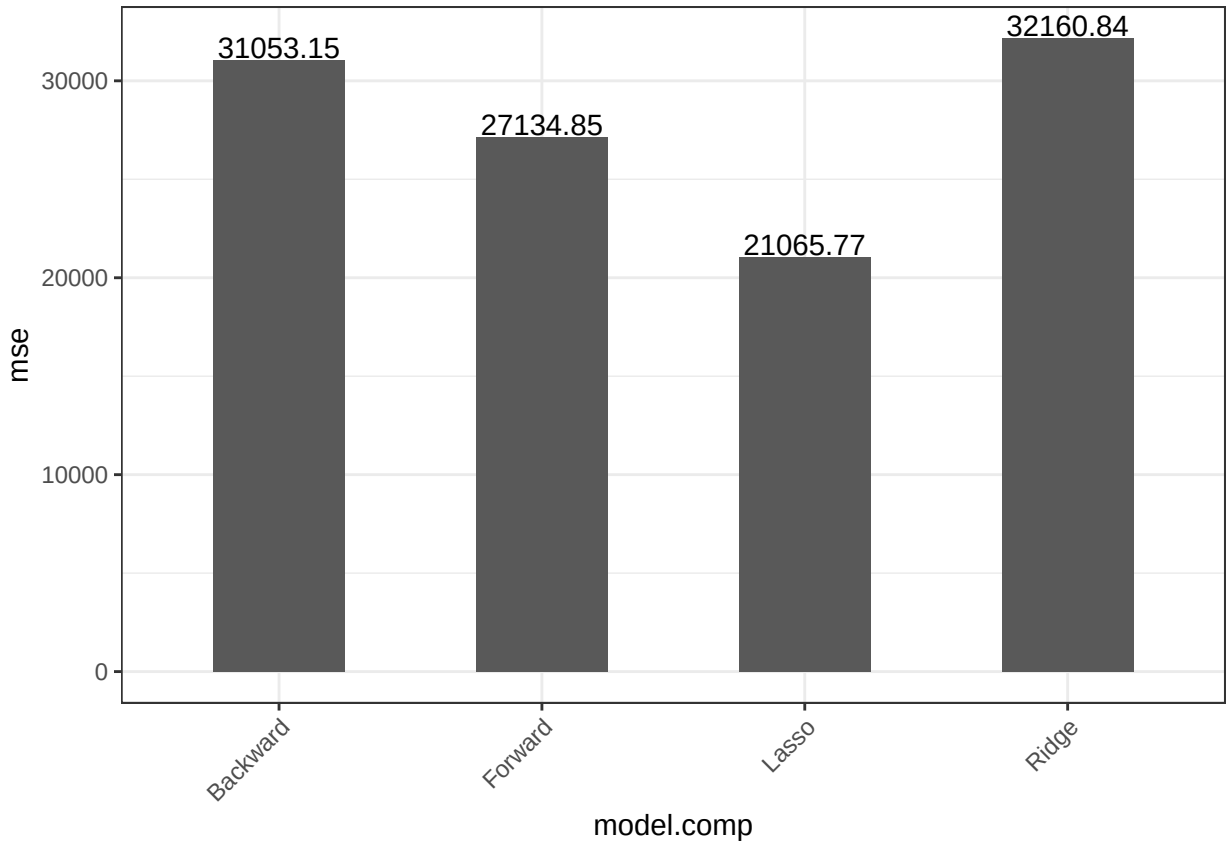
```
## [1] "Training MSE error: 16415.5041874492"
```

```
#Test predictions: using training model
pred.test.lasso <- predict(mod.lasso.train, newx = test.mat)

test.lasso.mse <- mean((pred.test.lasso - Test$price_cents)^2)
paste("Test MSE error:",test.lasso.mse)
```

```
## [1] "Test MSE error: 21065.7697663407"
```

Comparing results

```
df_compar <- data.frame(model.comp = c("Forward", "Backward", "Ridge", "Lasso"),

mse = c(test.for.mse, test.bck.mse, test.ridge.mse, test.lasso.mse))

ggplot(data = df_compar, aes(x = model.comp, y = mse)) + geom_col(width = 0.5) +
    geom_text(aes(label = round(mse, 2)), vjust = -0.1) + theme_bw() + theme(axis.text.x = element_text
    hjust = 1))
```



Linear Model

```
lmodel.Train <- lm (price_cents ~ `area squared foot` + Locker + Unit + `Air con` + `24 hours` + `Lift a

summary(lmodel.Train)
```

```
##
## Call:
## lm(formula = price_cents ~ `area squared foot` + Locker + Unit +
##     `Air con` + `24 hours` + `Lift access` + `Loading dock` +
##     `Step ladder` + `Racking/Shelving`, data = Training)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -397.61  -59.65    0.37   67.28  447.65
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)            741.702     99.604   7.447 3.97e-12 ***
## `area squared foot`     54.865      2.541  21.589  < 2e-16 ***
## Locker                -245.022     39.080  -6.270 2.66e-09 ***
## Unit                    57.726     50.969   1.133 0.258914
## `Air con`              107.085     57.878   1.850 0.065942 .
```

```
## `24 hours`          -200.498     80.017  -2.506 0.013119 *
## `Lift access`       -304.919     99.828  -3.054 0.002601 **
## `Loading dock`        64.024     17.844   3.588 0.000431 ***
## `Step ladder`        135.079     46.819   2.885 0.004396 **
## `Racking/Shelving`   130.413     94.549   1.379 0.169528
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 114 on 178 degrees of freedom
## Multiple R-squared:  0.8004, Adjusted R-squared:  0.7903
## F-statistic: 79.33 on 9 and 178 DF,  p-value: < 2.2e-16
```

```
lmodel.Test <- lm (price_cents ~ `area squared foot` + Locker + Unit + `Air con` + `24 hours` + `Lift a

summary(lmodel.Test)
```

```
##
## Call:
## lm(formula = price_cents ~ `area squared foot` + Locker + Unit +
##     `Air con` + `24 hours` + `Lift access` + `Loading dock` +
##     `Step ladder` + `Racking/Shelving`, data = Test)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -279.20  -57.42   -0.93   74.03  350.26
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)         1005.799    203.952   4.932 5.80e-06 ***
## `area squared foot`   59.998      4.774  12.567  < 2e-16 ***
## Locker              -156.905     68.594  -2.287 0.025383 *
## Unit                  36.098     80.391   0.449 0.654883
## `Air con`             56.174     90.733   0.619 0.537973
## `24 hours`          -296.252    145.523  -2.036 0.045790 *
## `Lift access`       -491.882    134.603  -3.654 0.000512 ***
## `Loading dock`         2.359     29.241   0.081 0.935938
## `Step ladder`        349.811     83.554   4.187 8.56e-05 ***
## `Racking/Shelving`   114.807    100.826   1.139 0.258963
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 121.4 on 66 degrees of freedom
## Multiple R-squared:  0.8115, Adjusted R-squared:  0.7858
## F-statistic: 31.57 on 9 and 66 DF,  p-value: < 2.2e-16
```