House Prices

```
library(data.table, quietly = TRUE)
library(dplyr, quietly = TRUE)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:data.table':
##
##     between, first, last

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(caret, quietly = TRUE)
library(ggplot2, quietly = TRUE)
```

Đọc dữ liệu

```
# Data import
raw.test <- fread(input = "test.csv", sep = ",", stringsAsFactors = F, data.table = F)
raw.train <- fread(input = "train.csv", sep = ",", stringsAsFactors = F, data.table = F)
# Kết hợp 2 tập data thành 1 và gắn nhãn để tiện cho việc tiền xử lý
fulldt <- rbind(raw.train[,-81], raw.test)
fulldt <- cbind(fulldt, Set = c(rep("Train", times = dim(raw.train)[1]),
                                rep("Test", times = dim(raw.test)[1])))
```

Tiền xử lý dữ liệu

Thay thế những dữ liệu bị thiếu

```
# Kiểm tra những dữ liệu nào bị NA
x <- colSums(sapply(fulldt, is.na))
# Tạo data frame
```

```
x <- data.frame(Variables = names(x), NA.Count = x); rownames(x) <- c()
# Loại bỏ những biến nào không có giá trị NA
x <- x %>% filter(NA.Count > 0)
x
```

```
##       Variables NA.Count
## 1      MSZoning        4
## 2   LotFrontage      486
## 3         Alley     2721
## 4     Utilities        2
## 5   Exterior1st        1
## 6   Exterior2nd        1
## 7    MasVnrType       24
## 8    MasVnrArea       23
## 9      BsmtQual       81
## 10     BsmtCond       82
## 11 BsmtExposure       82
## 12 BsmtFinType1       79
## 13   BsmtFinSF1        1
## 14 BsmtFinType2       80
## 15   BsmtFinSF2        1
## 16    BsmtUnfSF        1
## 17  TotalBsmtSF        1
## 18   Electrical        1
## 19 BsmtFullBath        2
## 20 BsmtHalfBath        2
## 21  KitchenQual        1
## 22   Functional        2
## 23  FireplaceQu     1420
## 24   GarageType      157
## 25  GarageYrBlt      159
## 26 GarageFinish      159
## 27   GarageCars        1
## 28   GarageArea        1
## 29   GarageQual      159
## 30   GarageCond      159
## 31       PoolQC     2909
## 32        Fence     2348
## 33  MiscFeature     2814
## 34     SaleType        1
```

Thay thế những giá trị bị thiếu bằng "None", tức là không có

```
y <- c("Alley", "BsmtQual", "BsmtCond", "BsmtExposure", "BsmtFinType1",
       "BsmtFinType2", "FireplaceQu", "GarageType", "GarageFinish",
       "GarageQual", "GarageCond", "PoolQC", "Fence", "MiscFeature")
fulldt[,y] <- apply(fulldt[,y], 2,
                    function(x) {
                            replace(x, is.na(x), "None")
                            }
                    )
```

Đối với biến số, để thể hiện "không có" thì ta sẽ dùng số 0 thay cho giá trị bị thiếu

```
y <- c("BsmtFinSF1", "BsmtFinSF2", "BsmtUnfSF", "TotalBsmtSF", "BsmtFullBath",
       "BsmtHalfBath", "GarageCars", "GarageArea", "MasVnrArea", "LotFrontage")
fulldt[,y] <- apply(fulldt[,y], 2,
                    function(x) {
                            replace(x, is.na(x), 0)
                            }
                    )
```

Thay thế những giá trị bị thiếu bằng giá trị xuất hiện nhiều nhất

```
y <- c("MasVnrType", "Electrical", "MSZoning", "Utilities", "Exterior1st",
       "Exterior2nd", "KitchenQual", "Functional", "SaleType")
fulldt[,y] <- apply(fulldt[,y], 2,
                    function(x) {
                            replace(x, is.na(x), names(which.max(table(x))))
                            }
                    )
```

Với biến GarageYrBlt (năm mà garage được xây dựng), ta sẽ gán nó cho một giá trị không có ý nghĩa

```
fulldt$GarageYrBlt[is.na(fulldt$GarageYrBlt)] <- -9999
```

Tạo thêm các biến tổng hợp mới

```
# Biến TotalBaths chứa tổng số lượng bồn tắm trong ngôi nhà
fulldt$TotalBaths <- fulldt$BsmtFullBath +
                                fulldt$BsmtHalfBath +
                                fulldt$FullBath +
                                fulldt$HalfBath

# Biến AreaAbvground chứ tổng diện tích của cả tầng 1 và tầng 2
fulldt$AreaAbvground <- fulldt$`1stFlrSF` + fulldt$`2ndFlrSF`

# Biến TotalArea chứa tổng diện tích tầng hầm và diện tích mặt đất
fulldt$TotalArea <- fulldt$GrLivArea + fulldt$TotalBsmtSF

# Biến TotalQual chứa điểm hoàn thiện và điểm tình trạng tổng thể của ngôi nhà
fulldt$TotalQual <- fulldt$OverallCond * fulldt$OverallQual
```

Sau khi tiền xử lý xong, ta tách tập dữ liệu thành 2 tập train test như ban đầu

```
train <- fulldt %>% filter(Set == "Train") %>% select(-Set) %>%
  cbind(SalePrice = raw.train$SalePrice)

test <- fulldt %>% filter(Set == "Test") %>% select(-Set)
```

Chọn mô hình Machine Learning phù hợp

```
set.seed(1)
```

Mô hình 1: Random forest cơ bản

Ta sử dụng Cross validation với `k = 5` cho toàn bộ các mô hình.

```
myControl = trainControl(method = "cv", number = 5, verboseIter = FALSE)
model_rf = train(SalePrice ~ .,
                 data = train,
                 tuneLength = 1,
                 method = "ranger",
                 importance = 'impurity',
                 trControl = myControl)
model_rf
```

```
## Random Forest
##
## 1460 samples
##   84 predictor
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 1169, 1166, 1169, 1167, 1169
## Resampling results across tuning parameters:
##
##   splitrule   RMSE      Rsquared   MAE
##   variance    28807.85  0.8774630  16582.21
##   extratrees  30864.73  0.8636822  17945.83
##
## Tuning parameter 'mtry' was held constant at a value of 16
## Tuning
##  parameter 'min.node.size' was held constant at a value of 5
## RMSE was used to select the optimal model using the smallest value.
## The final values used for the model were mtry = 16, splitrule = variance
##  and min.node.size = 5.
```

Các biến quan trọng

Hàm `varImp()` trả về danh sách 20 biến quan trọng nhất cho mô hình

```
varImp(model_rf)
```

```
## ranger variable importance
##
##   only 20 most important variables shown (out of 264)
##
##                Overall
## TotalArea       100.00
## OverallQual      87.75
## AreaAbvground    61.50
```

```
## GrLivArea       60.90
## TotalBsmtSF     52.33
## GarageArea      50.74
## `1stFlrSF`      41.76
## GarageCars      40.69
## ExterQualTA     37.15
## YearBuilt       35.22
## TotalBaths      33.89
## TotalQual       30.88
## GarageYrBlt     26.51
## KitchenQualTA   25.70
## BsmtFinSF1      22.86
## FullBath        22.43
## `2ndFlrSF`      22.00
## LotArea         19.45
## YearRemodAdd    18.56
## TotRmsAbvGrd    16.55
```

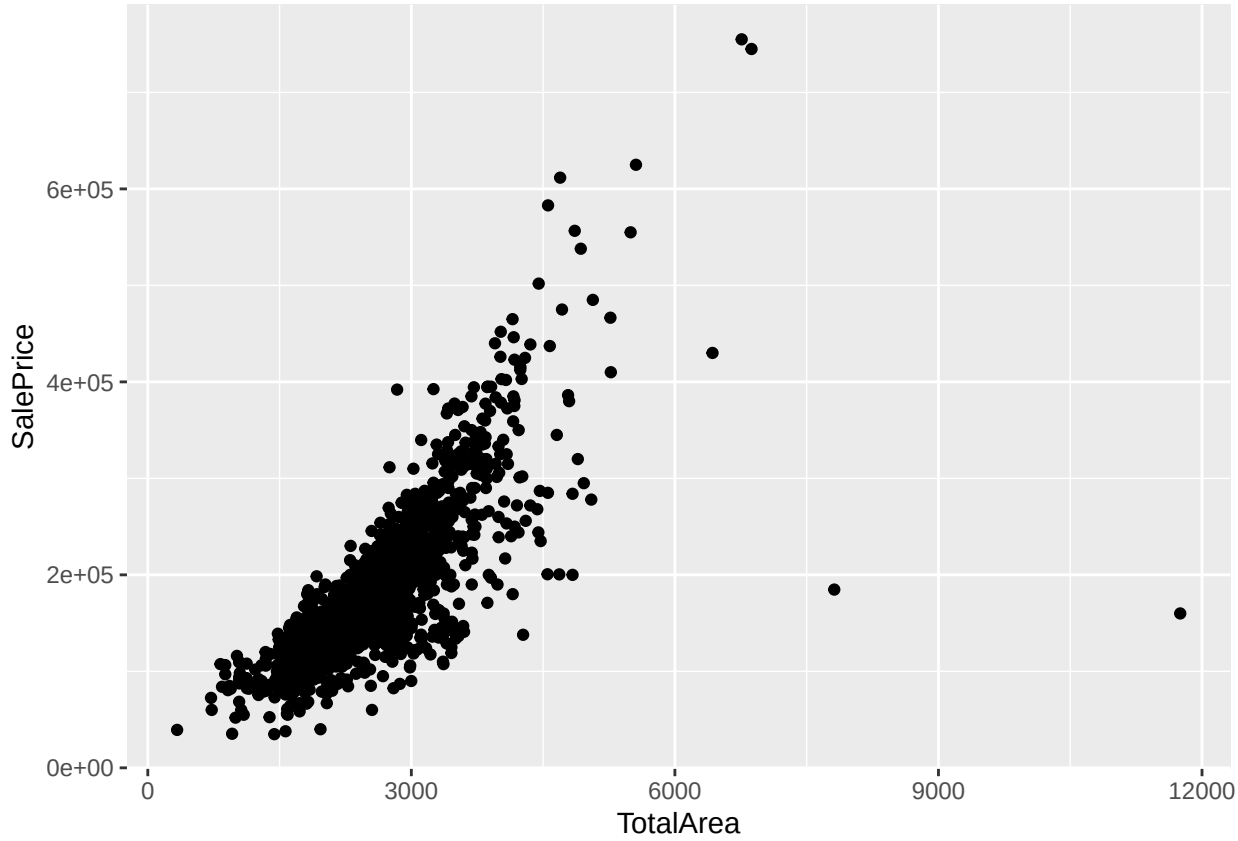
Loại bỏ outliers

Từ danh sách trên, ta tiến hành vẽ biểu đồ `scatter plot` giữa các biến đó với biến kết quả. Sau đó ta tiến hành loại bỏ outliers thủ công.

```
train_rmOulier <- train
```

Biến `TotalArea`

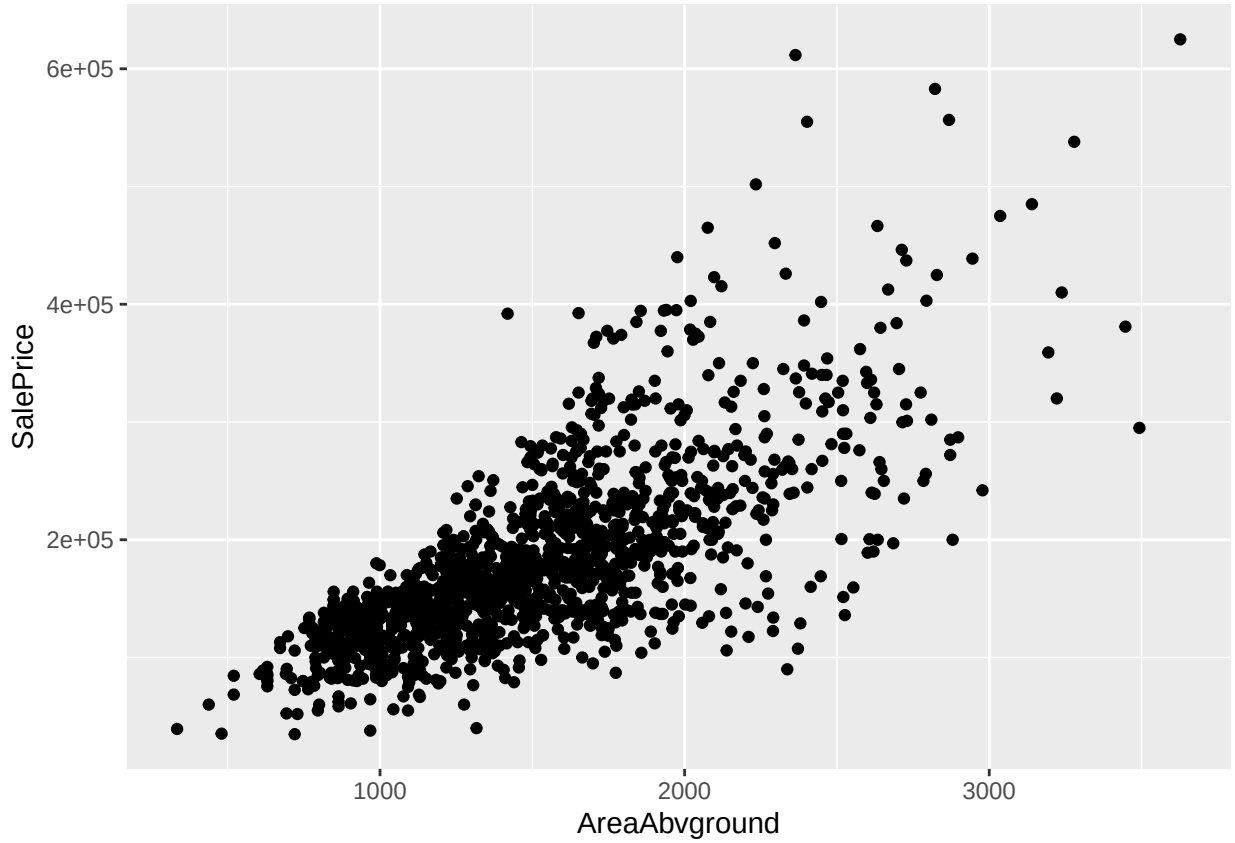
```
ggplot(train_rmOulier ,aes(y = SalePrice, x = TotalArea)) + geom_point()
```




```
train_rmOulier = filter(train_rmOulier, TotalArea <= 6000)
```

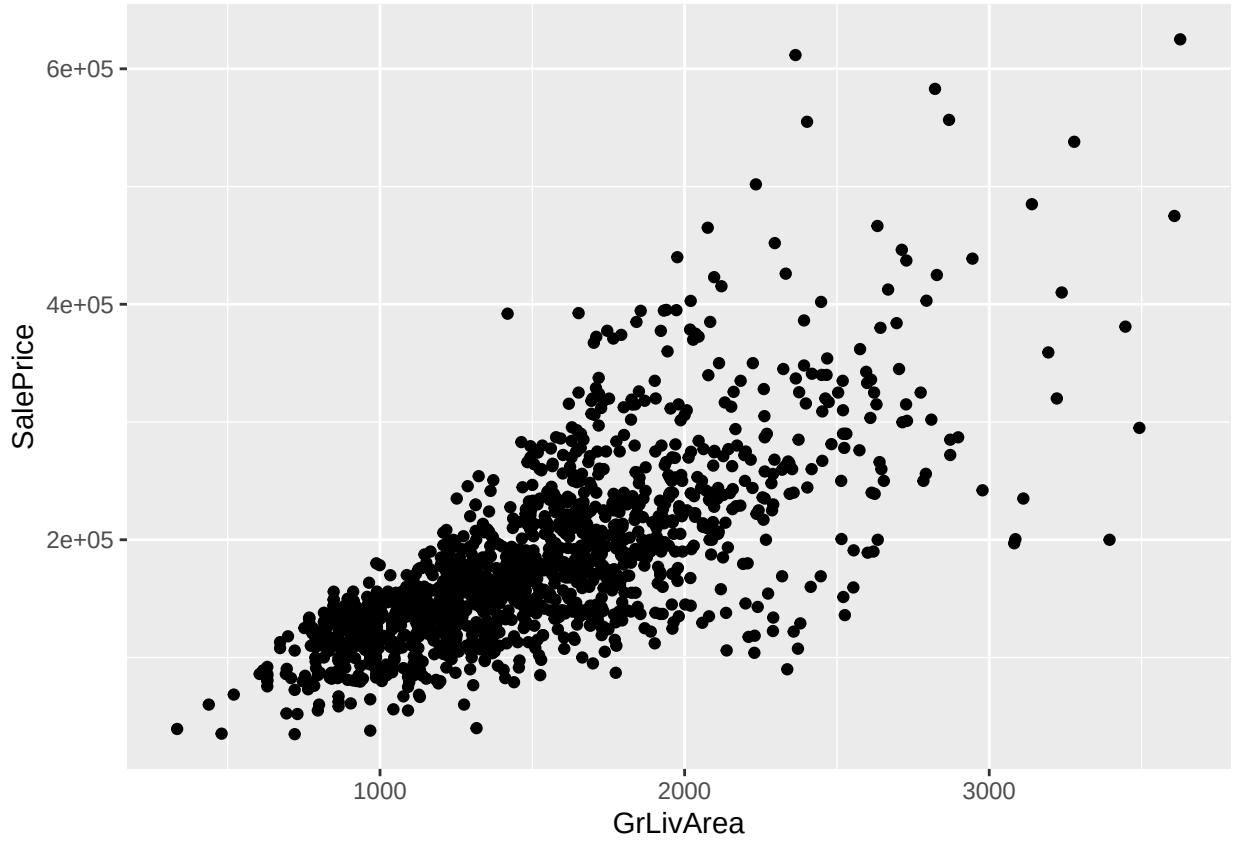
Biến `AreaAbvground`

```
ggplot(train_rmOulier, aes(y = SalePrice, x = AreaAbvground)) + geom_point()
```



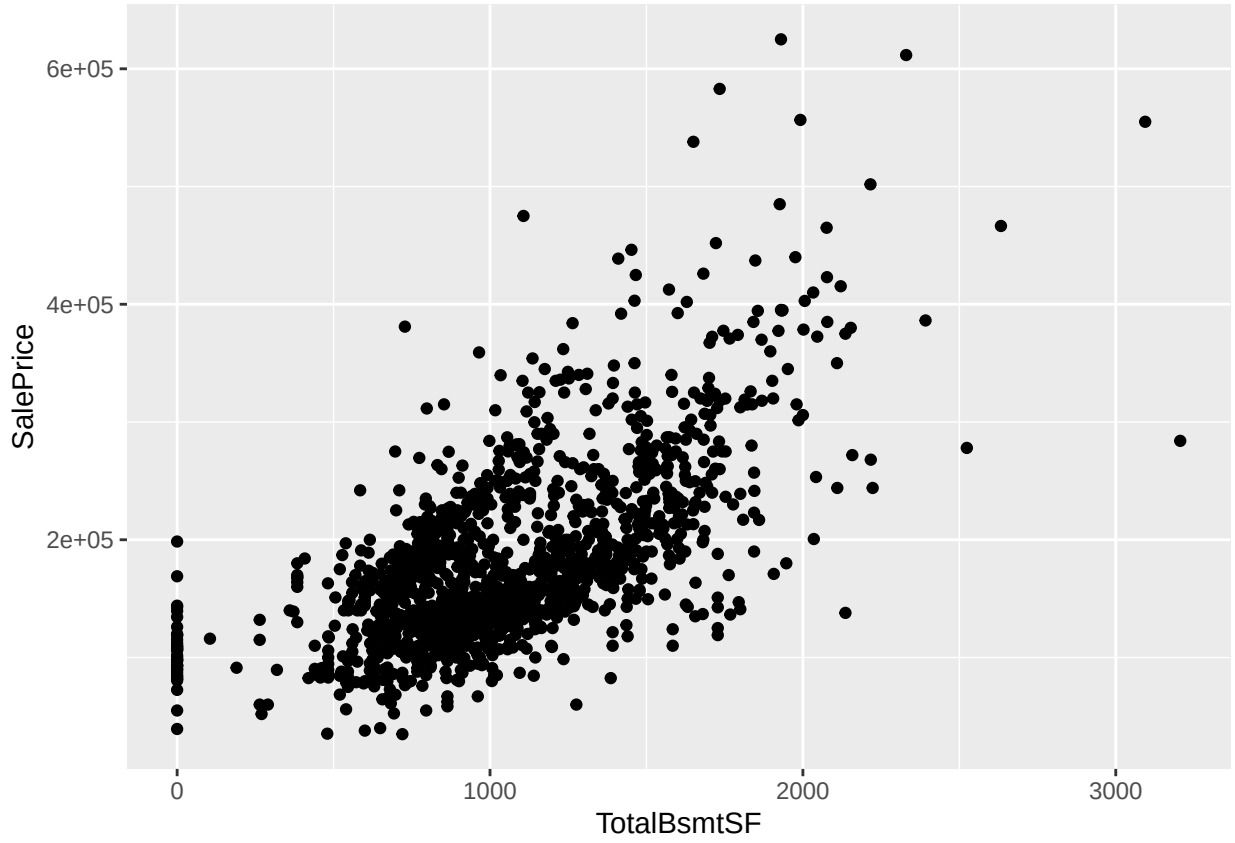
Biến `GrLivArea`

```
ggplot(train_rmOulier, aes(y = SalePrice, x = GrLivArea)) + geom_point()
```



Biến `TotalBsmtSF`

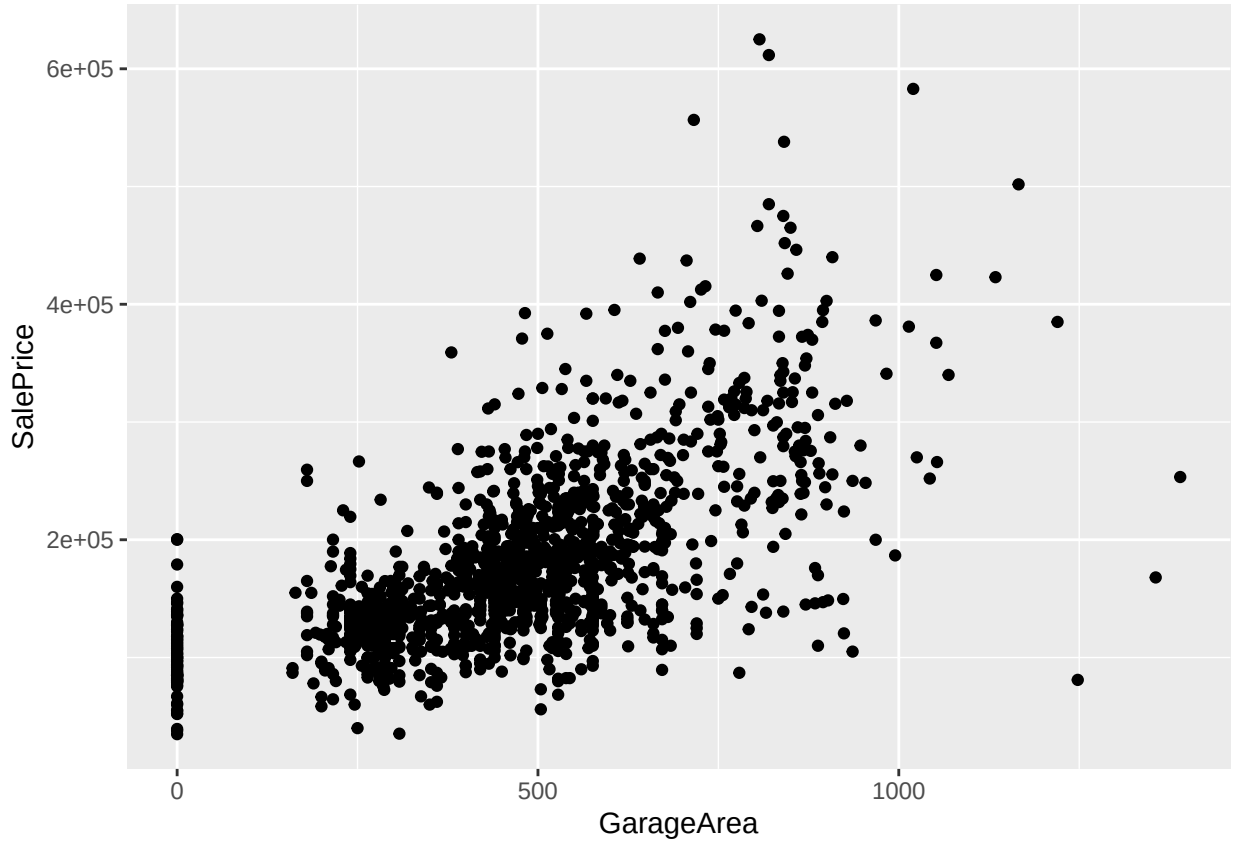
```
ggplot(train_rmOulier, aes(y = SalePrice, x = TotalBsmtSF)) + geom_point()
```




```
train_rmOulier = filter(train_rmOulier, TotalBsmtSF <= 3000)
```

Biến `GarageArea`

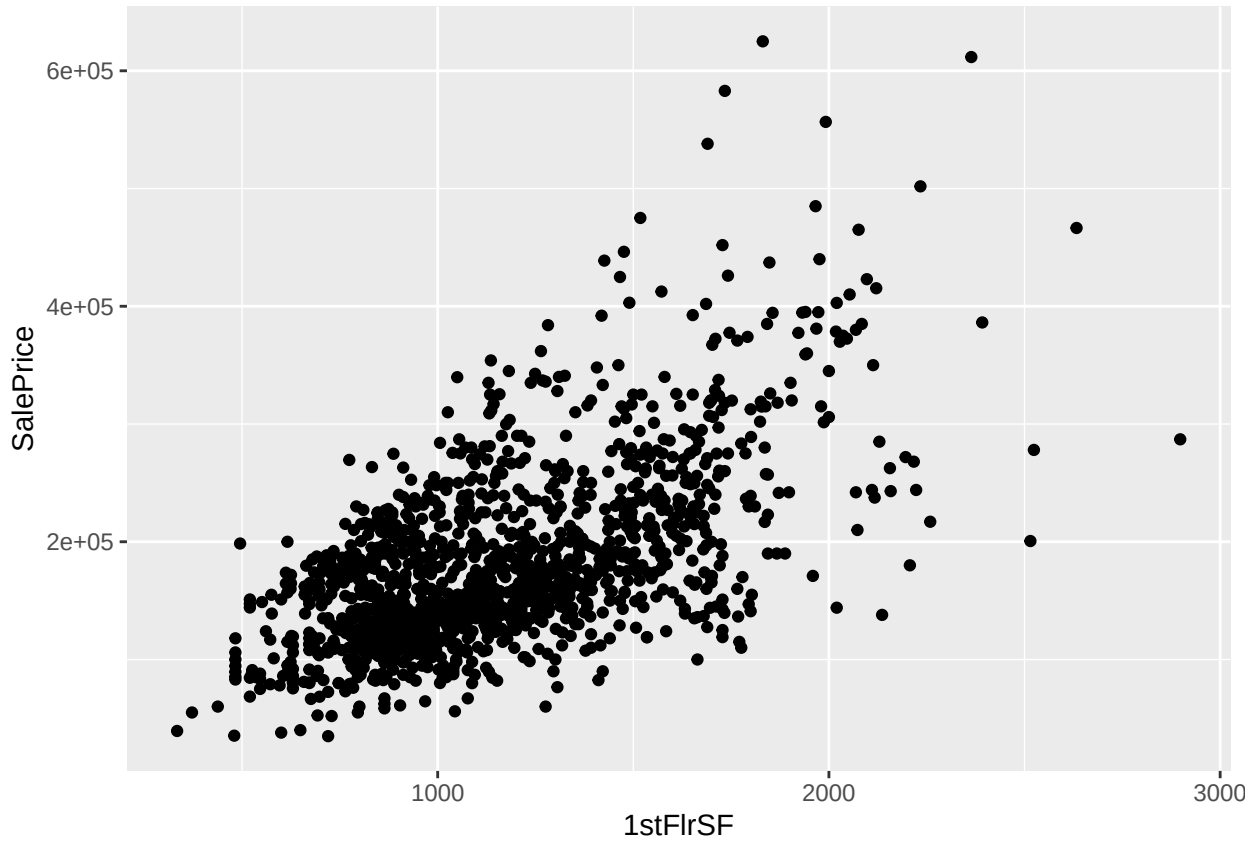
```
ggplot(train_rmOulier, aes(y = SalePrice, x = GarageArea)) + geom_point()
```




```
train_rmOulier = filter(train_rmOulier, GarageArea <= 1247)
```

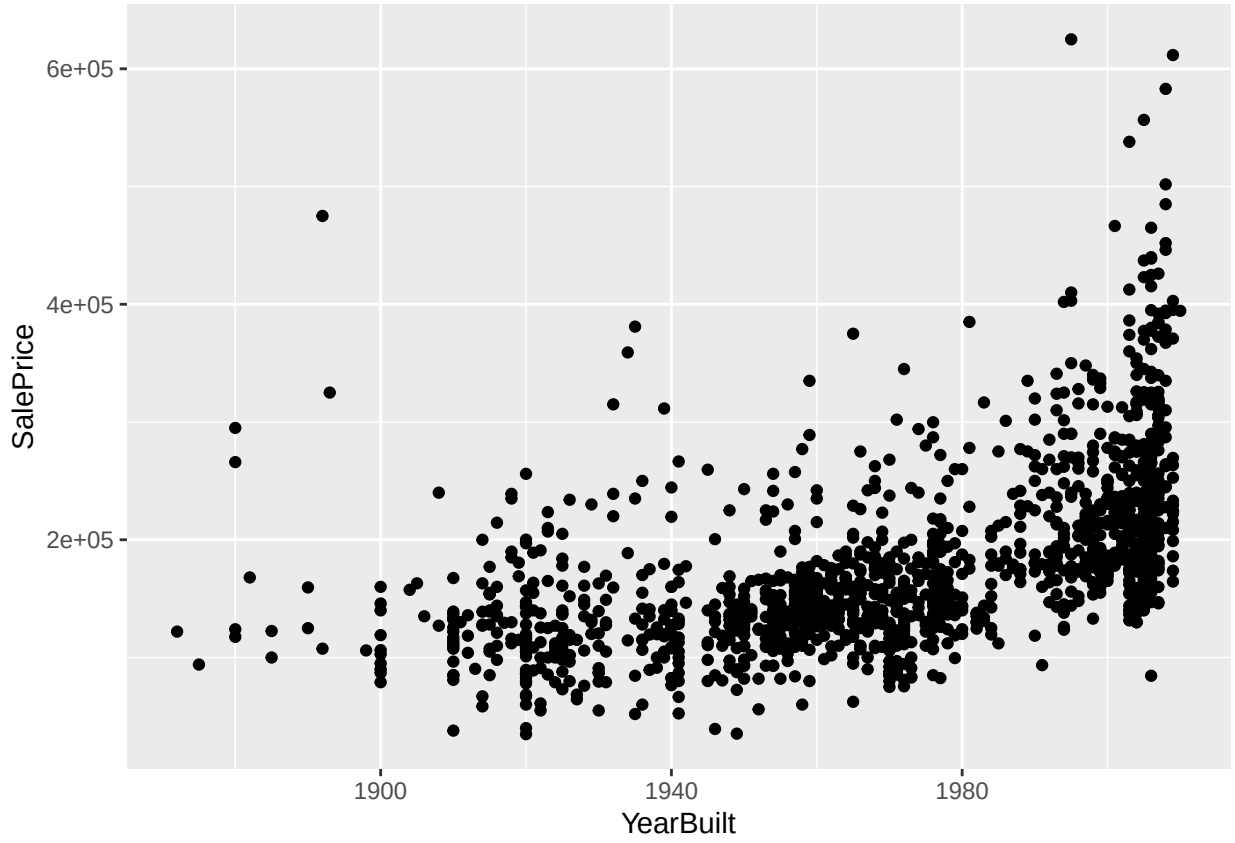
Biến `1stFlrSF`

```
ggplot(train_rmOulier, aes(y = SalePrice, x = `1stFlrSF`)) + geom_point()
```



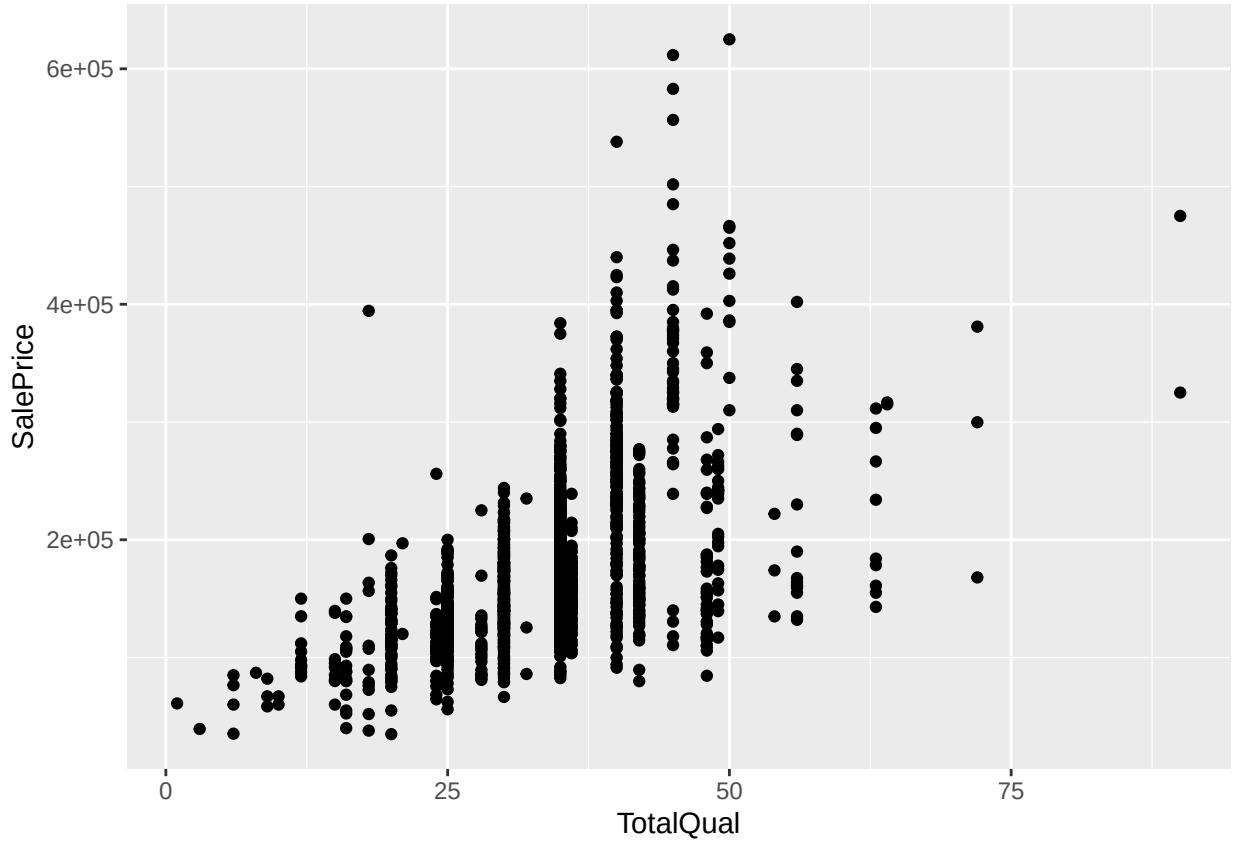
Biến `YearBuilt`

```
ggplot(train_rmOulier, aes(y = SalePrice, x = YearBuilt)) + geom_point()
```



Biến `TotalQual`

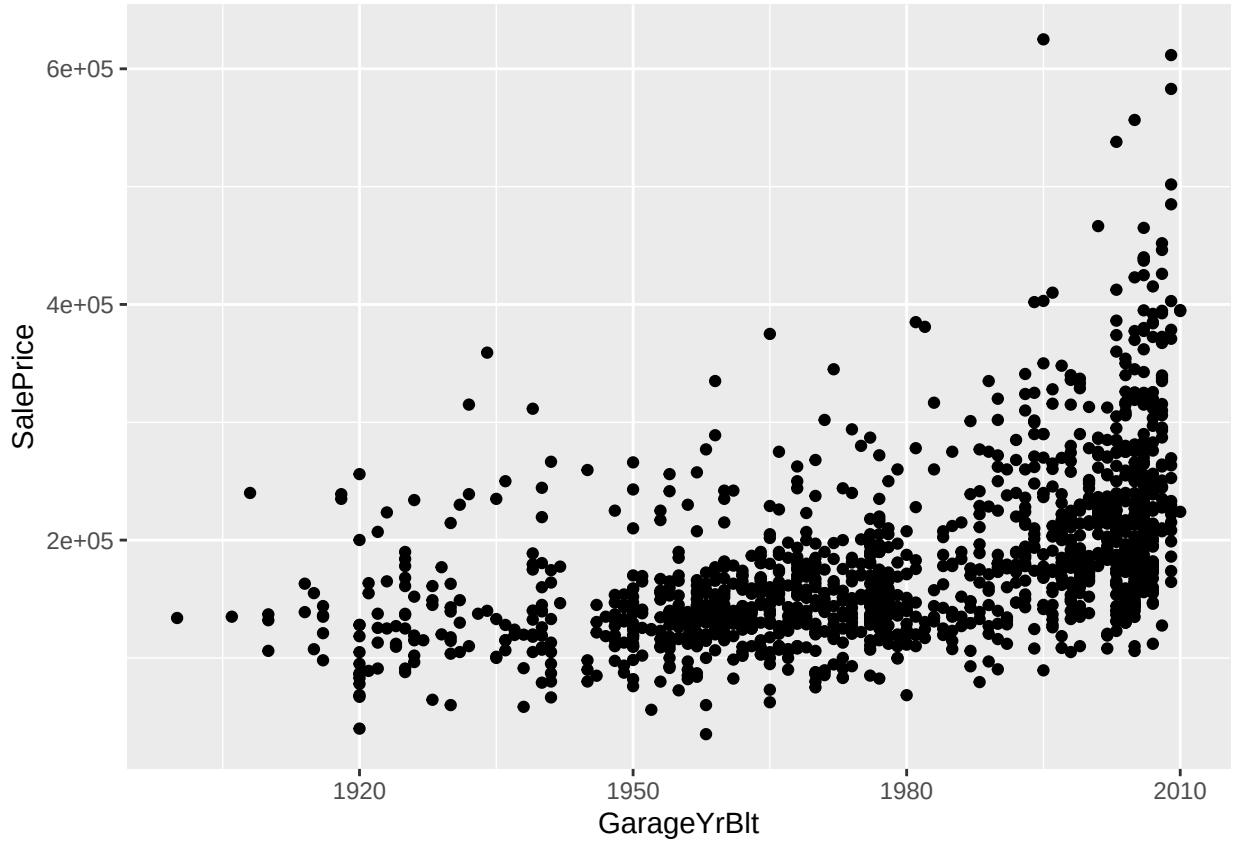
```
ggplot(train_rmOulier, aes(y = SalePrice, x = TotalQual)) + geom_point()
```




```
train_rmOulier <- filter(train_rmOulier, TotalQual <= 75)
```

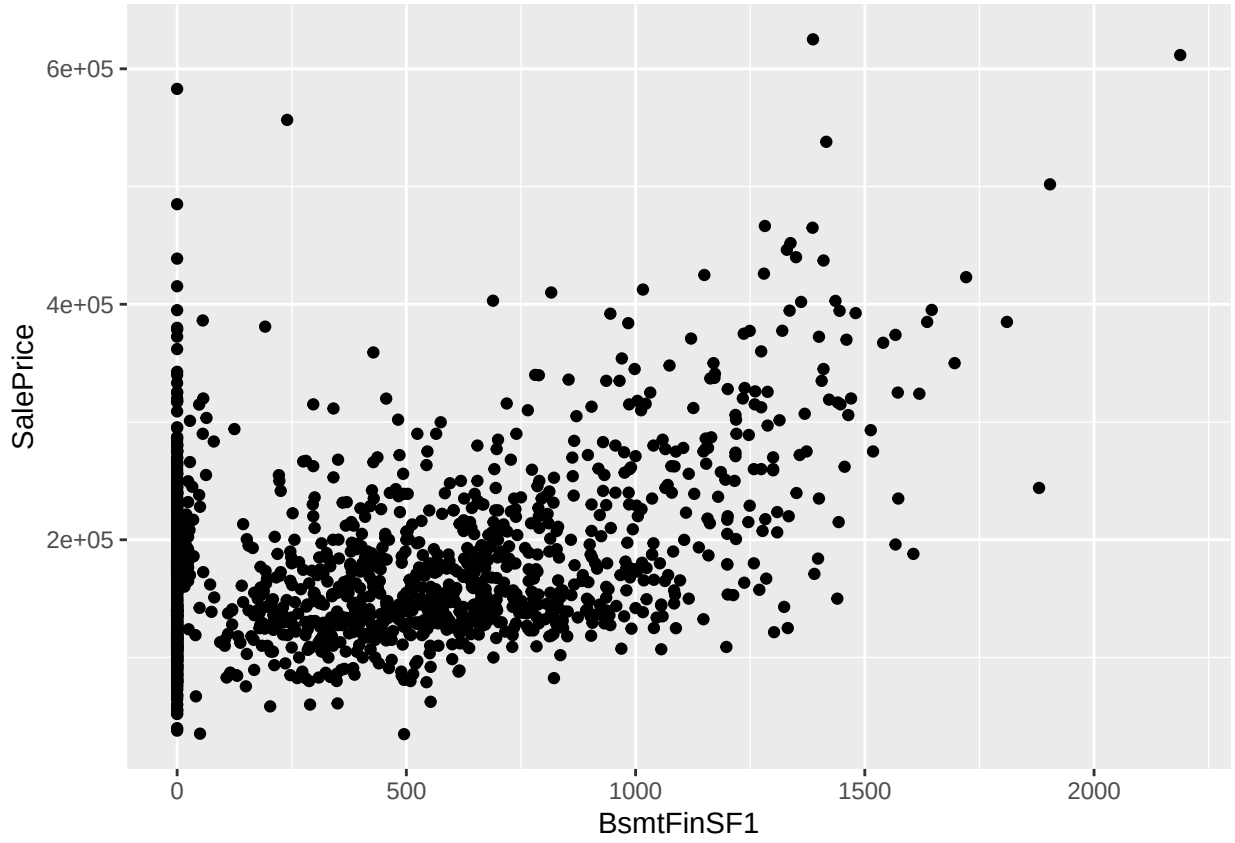
Biến `GarageYrBlt`

```
train_rmOulier %>% filter(GarageYrBlt > 0) %>%
  ggplot(aes(y = SalePrice, x = GarageYrBlt)) + geom_point()
```



Biến `BsmtFinSF1`

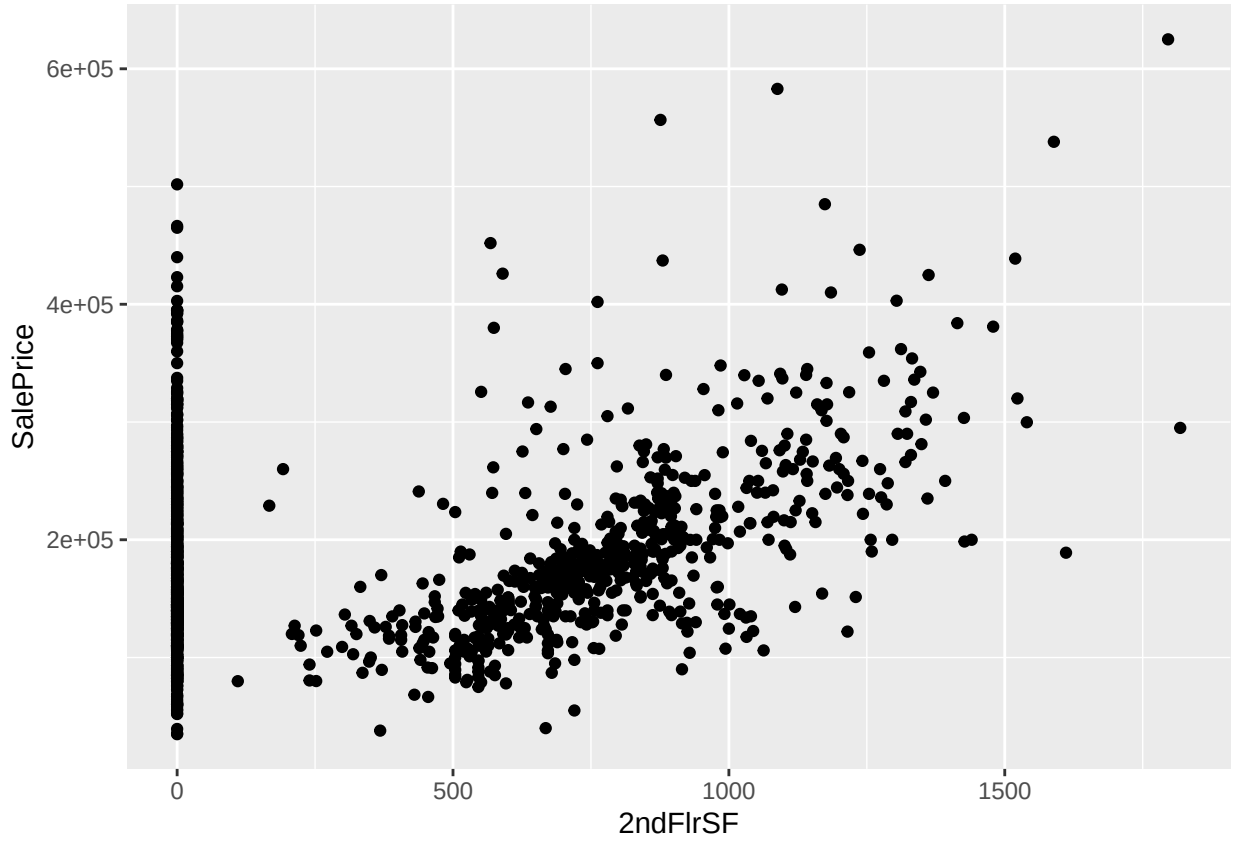
```
ggplot(train_rmOulier, aes(y = SalePrice, x = BsmtFinSF1)) + geom_point()
```




```
train_rmOulier <- filter(train_rmOulier, BsmtFinSF1 <= 2000)
```

Biến 2ndFlrSF

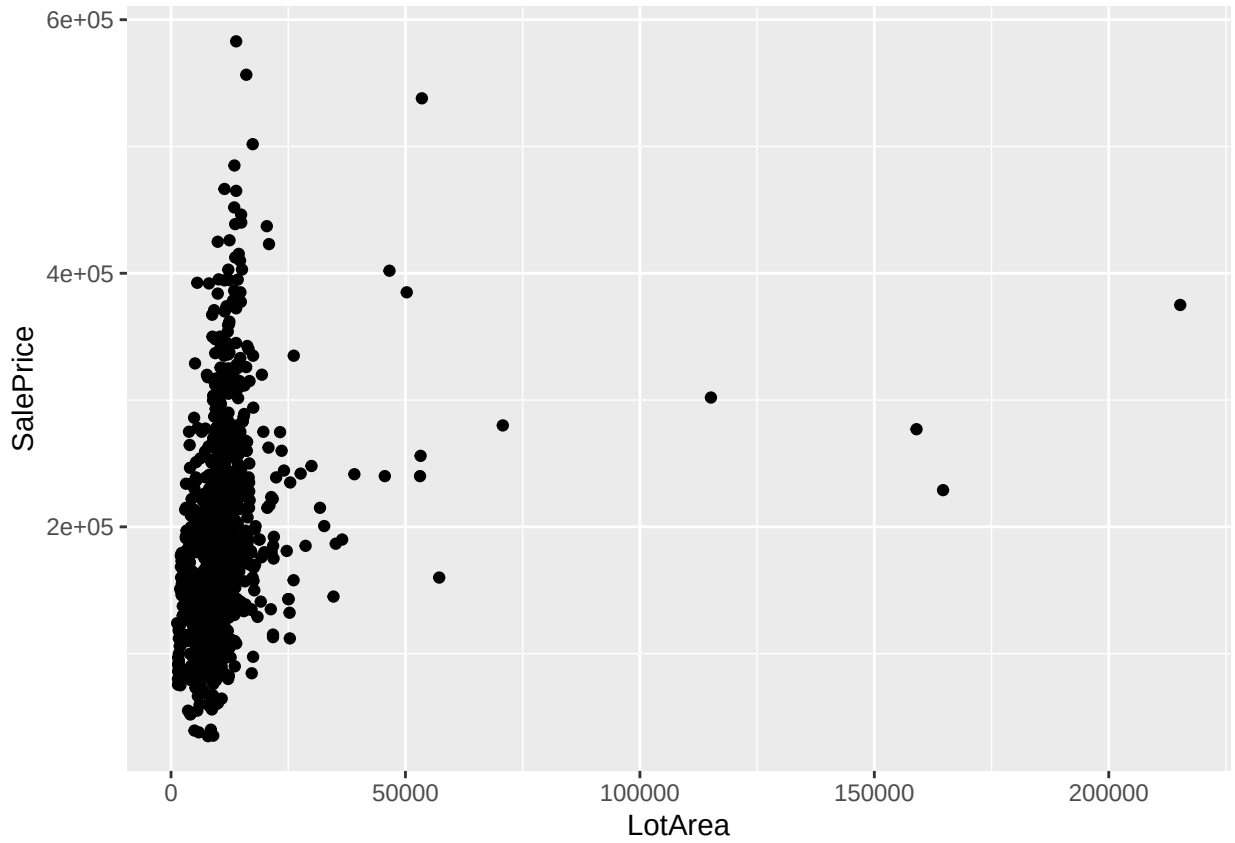
```
ggplot(train_rmOulier, aes(y = SalePrice, x = `2ndFlrSF`)) + geom_point()
```




```
train_rmOulier <- filter(train_rmOulier, `2ndFlrSF` <= 1750)
```

Biến `LotArea`

```
ggplot(train_rmOulier, aes(y = SalePrice, x = LotArea)) + geom_point()
```



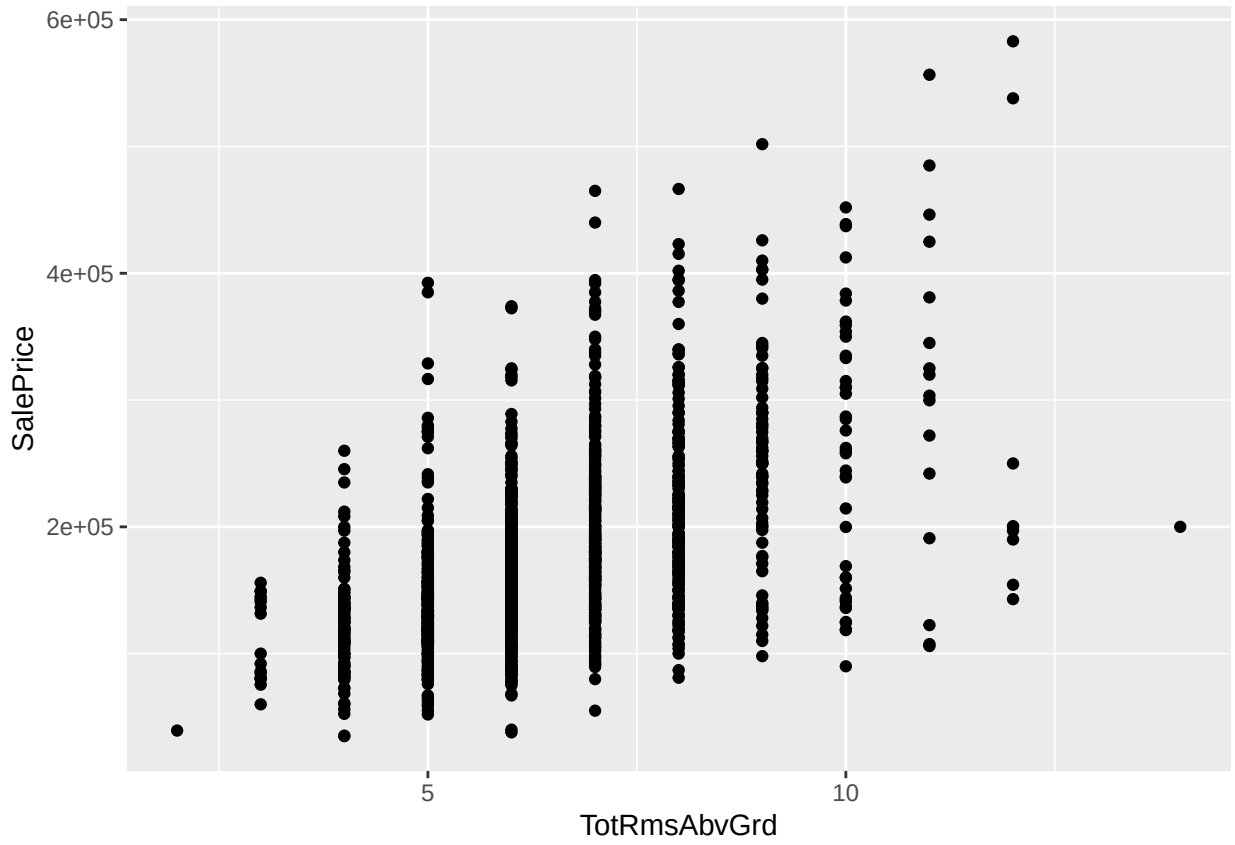

```
train_rmOulier <- filter(train_rmOulier, LotArea <= 100000)
```

Biến `YearRemodAdd`

```
ggplot(train_rmOulier, aes(y = SalePrice, x = YearRemodAdd)) + geom_point()
```

Biến `TotRmsAbvGrd`

```
ggplot(train_rmOulier, aes(y = SalePrice, x = TotRmsAbvGrd)) + geom_point()
```




```
train_rmOulier <- filter(train_rmOulier, TotRmsAbvGrd <= 12.5)
```

Mô hình 2: Random forest với dữ liệu đã loại bỏ outliers

Ta thử nghiệm lại mô hình `Random forest` trên tập dữ liệu đã loại bỏ outliers ở trên.

```
model_rf_rmo = train(SalePrice ~ .,
                data = train_rmOulier,
                tuneLength = 1,
                method = "ranger",
                importance = 'impurity',
                trControl = myControl)
model_rf_rmo
```

```
## Random Forest
##
## 1440 samples
##   84 predictor
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 1152, 1153, 1151, 1153, 1151
## Resampling results across tuning parameters:
##
```

```
##   splitrule   RMSE      Rsquared   MAE
##   variance    23535.04  0.9097845  15422.18
##   extratrees  25153.98  0.8998382  16505.31
##
## Tuning parameter 'mtry' was held constant at a value of 16
## Tuning
##  parameter 'min.node.size' was held constant at a value of 5
## RMSE was used to select the optimal model using the smallest value.
## The final values used for the model were mtry = 16, splitrule = variance
##  and min.node.size = 5.
```

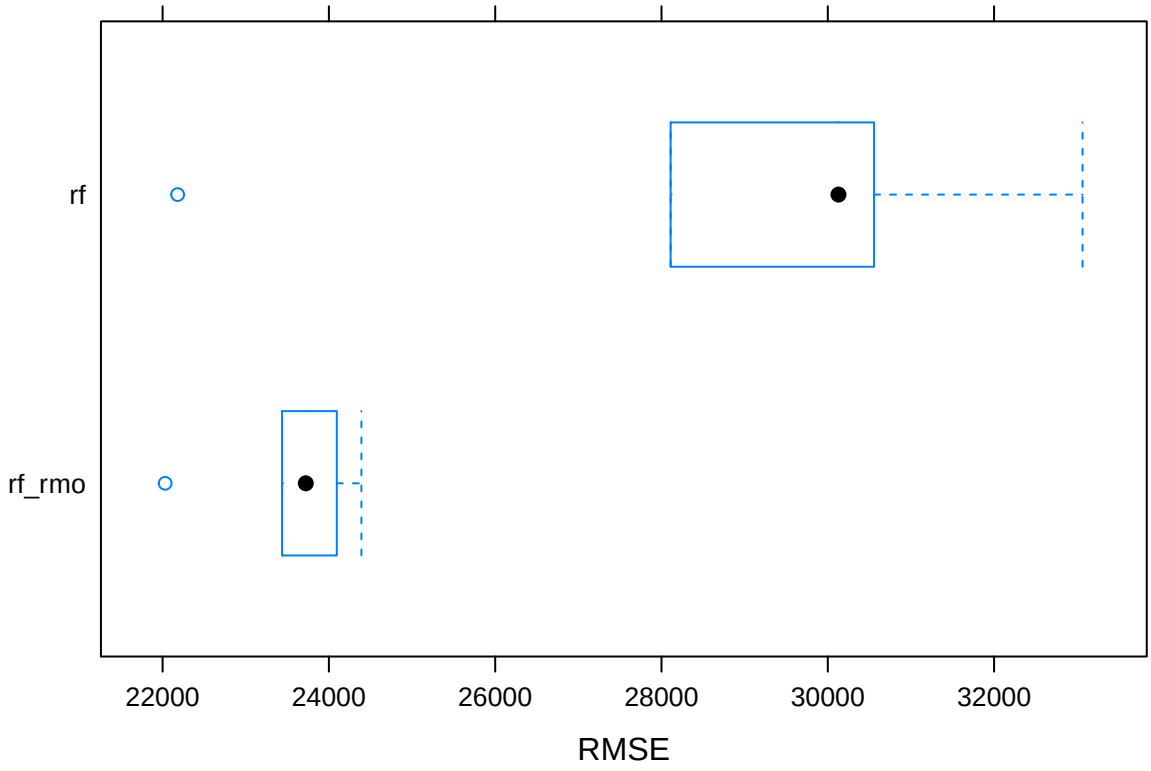
So sánh hiệu suất của 2 mô hình trên

```
model_list <- list(rf = model_rf, rf_rmo = model_rf_rmo)
resamples = resamples(model_list)
summary(resamples)
```

```
##
## Call:
## summary.resamples(object = resamples)
##
## Models: rf, rf_rmo
## Number of resamples: 5
##
## MAE
##            Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
## rf     14827.28 16565.51 16601.03 16582.21 16805.97 18111.25    0
## rf_rmo 14726.97 14798.63 15415.63 15422.18 15888.26 16281.43    0
##
## RMSE
##            Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
## rf     22180.82 28109.86 30128.31 28807.85 30556.34 33063.95    0
## rf_rmo 22030.64 23436.61 23722.08 23535.04 24095.13 24390.73    0
##
## Rsquared
##             Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## rf     0.8137256 0.8421746 0.8924892 0.8774630 0.9157452 0.9231806    0
## rf_rmo 0.9018922 0.9021466 0.9116226 0.9097845 0.9130597 0.9202015    0
```

Biểu diễn dưới dạng biểu đồ

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




```
rm(resamples, model_list)
```

Từ biểu đồ trên ta có thể thấy, việc loại bỏ outliers khiến cho sai số mô hình giảm đi đáng kể. Vậy nên ta sẽ áp dụng tập dữ liệu `train_rmOulier` cho những mô hình sau này.

Mô hình 3: Random forest với 20 biến quan trọng nhất

Về ý tưởng, ta sử dụng danh sách 20 biến quan trọng nhất ở trên để chạy thuật toán Random forest, giúp cho các cây con có thể được chọn những biến có giá trị quan trọng.

```
Top20Variables = c("TotalArea", "OverallQual", "AreaAbvground", "GrLivArea", "TotalBsmtSF",
                   "GarageArea", "1stFlrSF", "GarageCars", "ExterQual", "YearBuilt",
                   "TotalBaths", "TotalQual", "GarageYrBlt",
                   "KitchenQual", "BsmtFinSF1", "FullBath", "2ndFlrSF",
                   "LotArea", "YearRemodAdd", "TotRmsAbvGrd")
train_Top20Var = select(train_rmOulier, one_of(Top20Variables, "SalePrice"))

model_rf_Top20 = train(SalePrice ~ .,
                  data = train_Top20Var,
                  tuneLength = 1,
                  method = "ranger",
                  importance = 'impurity',
                  trControl = myControl)
model_rf_Top20
```

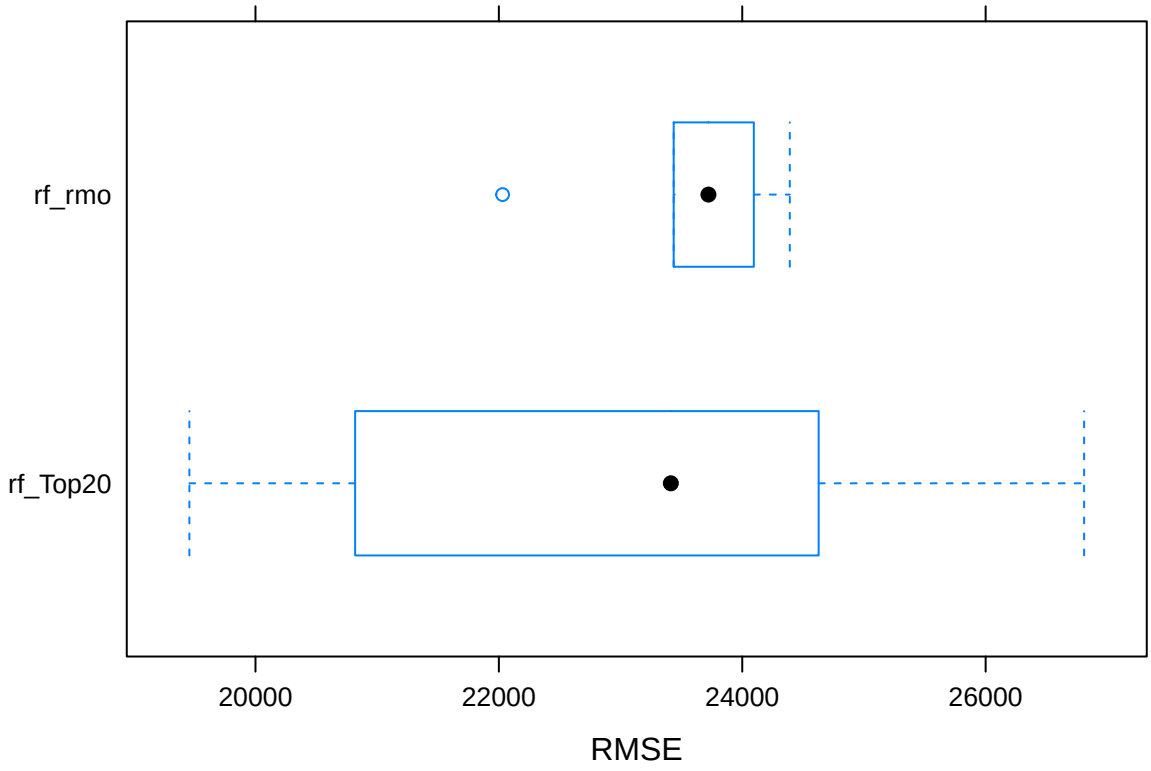
```
## Random Forest
##
## 1440 samples
##   20 predictor
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 1152, 1152, 1152, 1152, 1152
## Resampling results across tuning parameters:
##
##   splitrule   RMSE      Rsquared   MAE
##   variance    23025.18  0.9049154  15640.58
##   extratrees  23533.91  0.9018666  16022.64
##
## Tuning parameter 'mtry' was held constant at a value of 4
## Tuning
##  parameter 'min.node.size' was held constant at a value of 5
## RMSE was used to select the optimal model using the smallest value.
## The final values used for the model were mtry = 4, splitrule = variance
##  and min.node.size = 5.
```

So sánh mô hình này với mô hình Random forest cơ bản (mô hình đã loại bỏ outliers)

```
model_list = list(rf_rmo = model_rf_rmo, rf_Top20 = model_rf_Top20)
resamples = resamples(model_list)
summary(resamples)
```

```
##
## Call:
## summary.resamples(object = resamples)
##
## Models: rf_rmo, rf_Top20
## Number of resamples: 5
##
## MAE
##              Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
## rf_rmo   14726.97 14798.63 15415.63 15422.18 15888.26 16281.43    0
## rf_Top20 14148.13 15166.75 15776.42 15640.58 16132.27 16979.31    0
##
## RMSE
##              Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
## rf_rmo   22030.64 23436.61 23722.08 23535.04 24095.13 24390.73    0
## rf_Top20 19456.96 20818.90 23413.08 23025.18 24627.73 26809.24    0
##
## Rsquared
##               Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## rf_rmo   0.9018922 0.9021466 0.9116226 0.9097845 0.9130597 0.9202015    0
## rf_Top20 0.8919215 0.8982338 0.9054956 0.9049154 0.9062463 0.9226796    0
```

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




```
rm(resamples, model_list)
```

Ta thấy rằng mô hình `rf_Top20` hiệu quả hơn so với mô hình Random forest cơ bản

Mô hình 4: Hồi quy tuyến tính cơ bản

```
model_lm = train(SalePrice ~ .,
                 data = train_rmOulier,
                 method = "lm",
                 trControl = myControl)
```

```
## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient fit
## may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient fit
## may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient fit
## may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient fit
## may be misleading
```

```
## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient fit
## may be misleading
```

```
model_lm
```

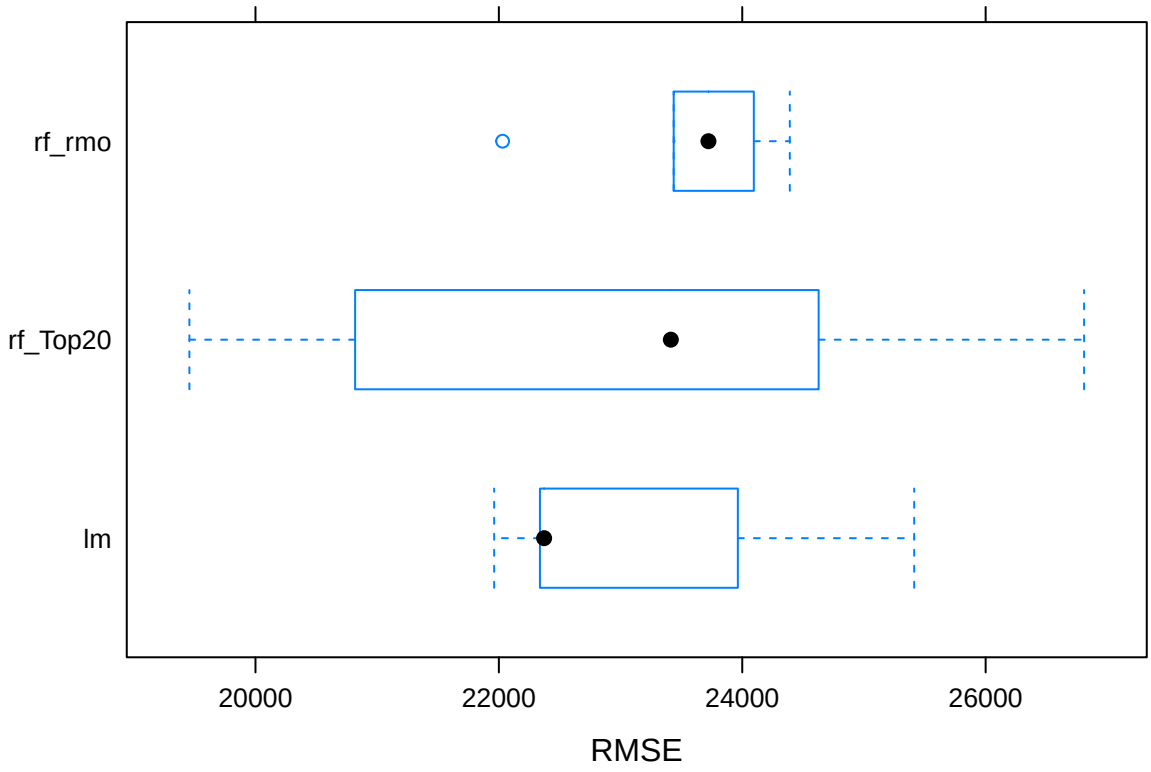
```
## Linear Regression
##
## 1440 samples
##   84 predictor
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 1152, 1153, 1152, 1151, 1152
## Resampling results:
##
##   RMSE      Rsquared   MAE
##   23209.49  0.9023035  15921.49
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

So sánh mô hình hồi quy tuyến tính với mô hình `rf_Top20`

```
model_list <- list(rf_rmo = model_rf_rmo, rf_Top20 = model_rf_Top20, lm = model_lm)
resamples = resamples(model_list)
summary(resamples)
```

```
##
## Call:
## summary.resamples(object = resamples)
##
## Models: rf_rmo, rf_Top20, lm
## Number of resamples: 5
##
## MAE
##              Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
## rf_rmo   14726.97 14798.63 15415.63 15422.18 15888.26 16281.43    0
## rf_Top20 14148.13 15166.75 15776.42 15640.58 16132.27 16979.31    0
## lm       15375.63 15417.30 15902.36 15921.49 15948.04 16964.14    0
##
## RMSE
##              Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
## rf_rmo   22030.64 23436.61 23722.08 23535.04 24095.13 24390.73    0
## rf_Top20 19456.96 20818.90 23413.08 23025.18 24627.73 26809.24    0
## lm       21961.55 22337.41 22371.90 23209.49 23963.72 25412.86    0
##
## Rsquared
##               Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## rf_rmo   0.9018922 0.9021466 0.9116226 0.9097845 0.9130597 0.9202015    0
## rf_Top20 0.8919215 0.8982338 0.9054956 0.9049154 0.9062463 0.9226796    0
## lm       0.8746614 0.9033311 0.9069625 0.9023035 0.9084728 0.9180897    0
```

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




```
rm(resamples, model_list)
```

Mô hình này thể hiện sự hiệu quả trên tập dữ liệu giá nhà. Ta thử nghiệm thêm một vài mô hình hồi quy khác.

Mô hình 5: Generalized Linear Model

```
model_glm = train(SalePrice ~ .,
                      data = train_rmOulier,
                      method = "glm",
                      trControl = myControl)
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = ifelse(type
## == : prediction from a rank-deficient fit may be misleading

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = ifelse(type
## == : prediction from a rank-deficient fit may be misleading

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = ifelse(type
## == : prediction from a rank-deficient fit may be misleading
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = ifelse(type
## == : prediction from a rank-deficient fit may be misleading

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = ifelse(type
## == : prediction from a rank-deficient fit may be misleading
```

```
model_glm
```

```
## Generalized Linear Model
##
## 1440 samples
##   84 predictor
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 1152, 1152, 1151, 1152, 1153
## Resampling results:
##
##   RMSE      Rsquared   MAE
##   23718.13  0.8977465  16017.3
```

Mô hình 6: Support Vector Machines with Linear Kernel

```
model_svm = train(SalePrice ~ .,
                  data = train_rmOulier,
                  method = "svmLinear",
                  trControl = myControl)
```

```
## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.
```

```
model_svm
```

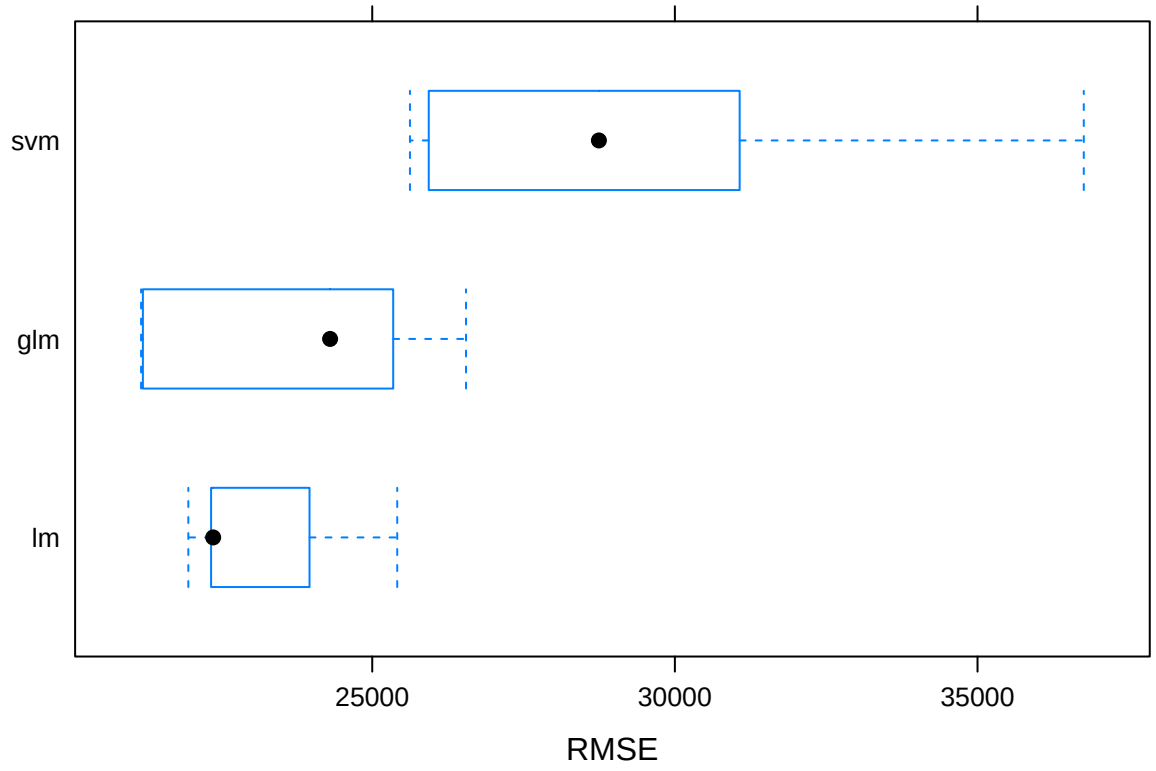
```
## Support Vector Machines with Linear Kernel
##
## 1440 samples
##   84 predictor
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 1153, 1152, 1152, 1151, 1152
## Resampling results:
```

```
##
##   RMSE      Rsquared  MAE
##   29625.33  0.84618   19400.21
##
## Tuning parameter 'C' was held constant at a value of 1
```

```
model_list <- list(lm = model_lm, svm = model_svm, glm = model_glm)
resamples = resamples(model_list)
summary(resamples)
```

```
##
## Call:
## summary.resamples(object = resamples)
##
## Models: lm, svm, glm
## Number of resamples: 5
##
## MAE
##         Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
## lm  15375.63 15417.30 15902.36 15921.49 15948.04 16964.14    0
## svm 17522.09 17924.67 19692.19 19400.21 20330.87 21531.22    0
## glm 15124.11 15203.41 16198.73 16017.30 16416.19 17144.07    0
##
## RMSE
##         Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
## lm  21961.55 22337.41 22371.90 23209.49 23963.72 25412.86    0
## svm 25622.99 25933.51 28744.84 29625.33 31069.88 36755.43    0
## glm 21181.18 21210.73 24303.65 23718.13 25346.22 26548.85    0
##
## Rsquared
##          Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## lm  0.8746614 0.9033311 0.9069625 0.9023035 0.9084728 0.9180897    0
## svm 0.8236094 0.8385774 0.8415352 0.8461800 0.8558377 0.8713403    0
## glm 0.8821797 0.8887155 0.8895874 0.8977465 0.9129350 0.9153147    0
```

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




```
rm(resamples, model_list)
```

Qua các thử nghiệm trên ta có thể thấy, hiện tại mô hình phù hợp nhất cho tập dữ liệu giá nhà là mô hình linear regression.