# Housing Price modeling - Linear Regression

*July 29, 2017*

```
library(tidyverse)
library(ggfortify)
library(glmnet)
library(GGally)
library(corrplot)
library(reshape2)
library(leaps)
```

## 1. Data Preparing

Loading data.

```
load("C:/Users/Bangda/Desktop/kaggle/housing-price/eda.RData")
needed_objects <- c(
  'test', 'test_na_filled', 'train', 'train_na_filled'
)
rm(list = setdiff(ls(), needed_objects))
```

Make a copy of data and split into `train` and `test`,

```
# Make copies
train_na_filled_copy <- train_na_filled
test_na_filled_copy  <- test_na_filled
# Append labels
label <- rep(1:5, length.out = nrow(train_na_filled))
```

Then we make a list of vairables might needed in modeling, we select those variables based on our work in EDA.

```
predictors <- c(
  'MSSubClass',   'MSZoning',    'LotFrontage', 'LotArea',      'Street',
  'LandContour',  'LandSlope',   'Neighborhood', 'Condition1',  'Condition2',
  'BldgType',     'HouseStyle',  'OverallQual', 'OverallCond',  'YearBuilt',
  'RoofStyle',    'Exterior1st', 'Exterior2nd', 'ExterQual',    'ExterCond',
  'BsmtCond',     'TotalBsmtSF', 'Heating',     'HeatingQC',    'CentralAir',
  'Electrical',   'GrLivArea',   'BedroomAbvGr', 'KitchenAbvGr', 'KitchenQual',
  'TotRmsAbvGrd', 'Fireplaces',  'FireplaceQu', 'GarageType',   'GarageYrBlt',
  'GarageFinish', 'GarageCars',  'GarageArea',  'GarageQual',   'GarageCond',
  'PavedDrive',   'MoSold',      'YrSold',      'SaleType',     'SaleCondition',
  'RoofMatl',     'MasVnrType',  'Foundation',  'BsmtQual'
)
```

## 2. Simple Linear Regression

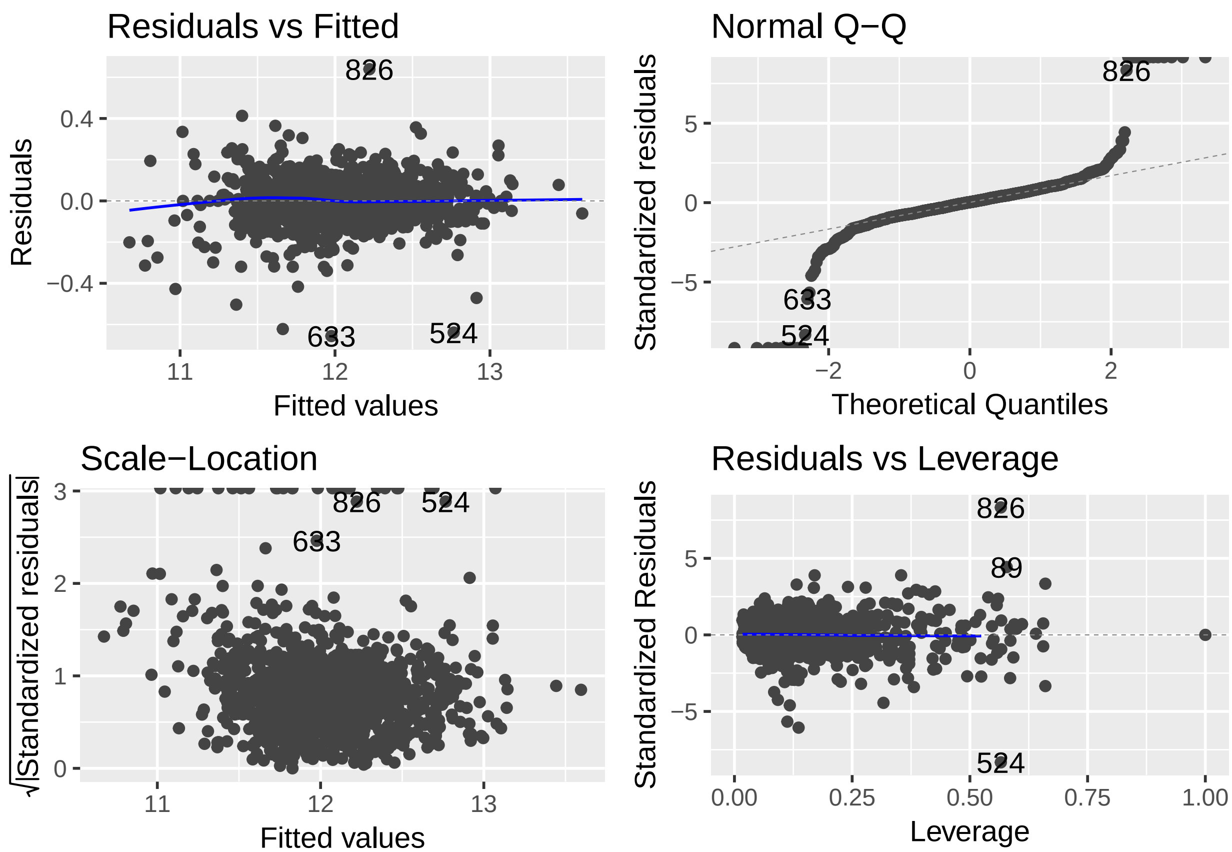
All predictors are simply included,

```
form1 <- paste(predictors, collapse = ' + ')
form1 <- formula(paste0('logSalePrice ~ ', form1))
```

```
linReg_model1 <- lm(form1, data = train_na_filled[label != 5, ])
linReg_model1_smy <- summary(linReg_model1)
linReg_model1_smy$r.squared
```

```
## [1] 0.9300552
```

```
ggplot2::autoplot(linReg_model1)
```




```
train_pred_linReg_model1 <- predict(linReg_model1,
                                    newdata = train_na_filled[label != 5, ])
mean((train_pred_linReg_model1 - train_na_filled[label != 5, ]$logSalePrice)^2) %>% sqrt()
```

```
## [1] 0.1066615
```

```
test_pred_linReg_model1 <- predict(linReg_model1,
                                   newdata = train_na_filled[label == 5, ])
mean((test_pred_linReg_model1 - train_na_filled[label == 5, ]$logSalePrice)^2) %>% sqrt()
```

```
## [1] 0.134218
```

```
form2 <- paste(predictors, collapse = ' + ')
form2 <- paste0(form2,
                '-BedroomAbvGr-Exterior1st-Exterior2nd-RoofStyle-BldgType-MSSubClass',
                '-ExterQual-ExterCond-Heating-FireplaceQu-YrSold-MasVnrType-Street-PavedDrive')
form2 <- formula(paste0('logSalePrice ~ ', form2))
linReg_model2 <- lm(form2, data = train_na_filled[label != 5, ])
linReg_model2_smy <- summary(linReg_model2)
linReg_model2_smy$r.squared
```

```
## [1] 0.9232585
```

```r
train_pred_linReg_model2 <- predict(linReg_model2,
                                    newdata = train_na_filled[label != 5, ])
mean((train_pred_linReg_model2 - train_na_filled[label != 5, ]$logSalePrice)^2) %>%
  sqrt()
```

```
## [1] 0.1117237
```

```r
test_pred_linReg_model2 <- predict(linReg_model2,
                                   newdata = train_na_filled[label == 5, ])
mean((test_pred_linReg_model2 - train_na_filled[label == 5, ]$logSalePrice)^2) %>%
  sqrt()
```

```
## [1] 0.1340385
```

Convert factors into numerical values.

```r
reArrangeOrder <- function(.variable) {
  if (class(train_na_filled[, .variable]) != 'character')
    return(train_na_filled[, .variable])

  orderFct <- train_na_filled %>%
    group_by_(.variable) %>%
    summarise(mid = median(SalePrice)) %>%
    arrange(mid)

  factor(train_na_filled[, .variable], levels = orderFct[[1]])
}
```

```r
for (i in 1:ncol(train_na_filled)) {
  train_na_filled[, i] <- as.numeric(reArrangeOrder(colnames(train_na_filled)[i]))
}
```

```r
form3 <- paste(predictors, collapse = ' + ')
form3 <- paste0(form3,
                '-BedroomAbvGr-Exterior1st-Exterior2nd-RoofStyle-BldgType-MSSubClass',
                '-ExterQual-ExterCond-Heating-FireplaceQu-YrSold-MasVnrType-Street-PavedDrive',
                '-LandSlope-KitchenAbvGr-GarageArea-GarageQual-SaleType')
form3 <- formula(paste0('logSalePrice ~ ', form3))
linReg_model3 <- lm(form3, data = train_na_filled[label != 5, ])
linReg_model3_smy <- summary(linReg_model3)
linReg_model3_smy$r.squared
```

```
## [1] 0.881078
```

```r
train_pred_linReg_model3 <- predict(linReg_model3,
                                    newdata = train_na_filled[label != 5, ])
mean((train_pred_linReg_model3 - train_na_filled[label != 5, ]$logSalePrice)^2) %>%
  sqrt()
```

```
## [1] 0.1390789
```

```r
test_pred_linReg_model3 <- predict(linReg_model3,
                                   newdata = train_na_filled[label == 5, ])
mean((test_pred_linReg_model3 - train_na_filled[label == 5, ]$logSalePrice)^2) %>%
  sqrt()
```

```
## [1] 0.1221187
```

## 3. Regularized Linear Regression

### (1) Ridge Regression

```
X <- as.matrix(train_na_filled[colnames(train_na_filled)%in%predictors])
y <- as.matrix(train_na_filled['logSalePrice'])
```

```
lambda <- 10^seq(10, -2, length = 100)
```

```
ridge_model1 <- glmnet(X[label != 5, ], y[label != 5],
                       alpha = 0, lambda = lambda)
ridge_cv_out1 <- cv.glmnet(X, y, alpha = 0)
best_lambda <- ridge_cv_out1$lambda.min
```

```
train_pred_ridge_model1 <- predict(ridge_model1, s = best_lambda,
                                   newx = X[label != 5, ])
mean((train_pred_ridge_model1 - y[label != 5])^2) %>% sqrt()
```

```
## [1] 0.1382493
```

```
test_pred_ridge_model1 <- predict(ridge_model1, s = best_lambda,
                                  newx = X[label == 5, ])
mean((test_pred_ridge_model1 - y[label == 5])^2) %>% sqrt()
```

```
## [1] 0.1206862
```

### (2) LASSO

```
lasso_model1 <- glmnet(X[label != 5, ], y[label != 5],
                       alpha = 1, lambda = lambda)
lasso_cv_out1 <- cv.glmnet(X, y, alpha = 1)
best_lambda <- lasso_cv_out1$lambda.min
```

```
train_pred_lasso_model1 <- predict(lasso_model1, s = best_lambda,
                                   newx = X[label != 5, ])
mean((train_pred_lasso_model1 - y[label != 5])^2) %>% sqrt()
```

```
## [1] 0.1424009
```

```
test_pred_lasso_model1 <- predict(lasso_model1, s = best_lambda,
                                  newx = X[label == 5, ])
mean((test_pred_lasso_model1 - y[label == 5])^2) %>% sqrt()
```

```
## [1] 0.1169136
```

```
lasso_coef1 <- predict(lasso_model1, type = 'coefficients',
        lambda = best_lambda)
# coef(lasso_cv_out1)
```

## 4. Improving Linear Regression

### (1) Simple Linear Regression

```
selected_predictors1 <- c(
  'MSZoning',     'Neighborhood', 'OverallQual', 'OverallCond',
```

```
  'BsmtQual',      'TotalBsmtSF',   'HeatingQC',    'CentralAir',
  'I(log(GrLivArea))',     'KitchenQual',   'Fireplaces',   'GarageType',
  'GarageFinish', 'GarageCars',    'GarageArea',   'SaleCondition'
)
```

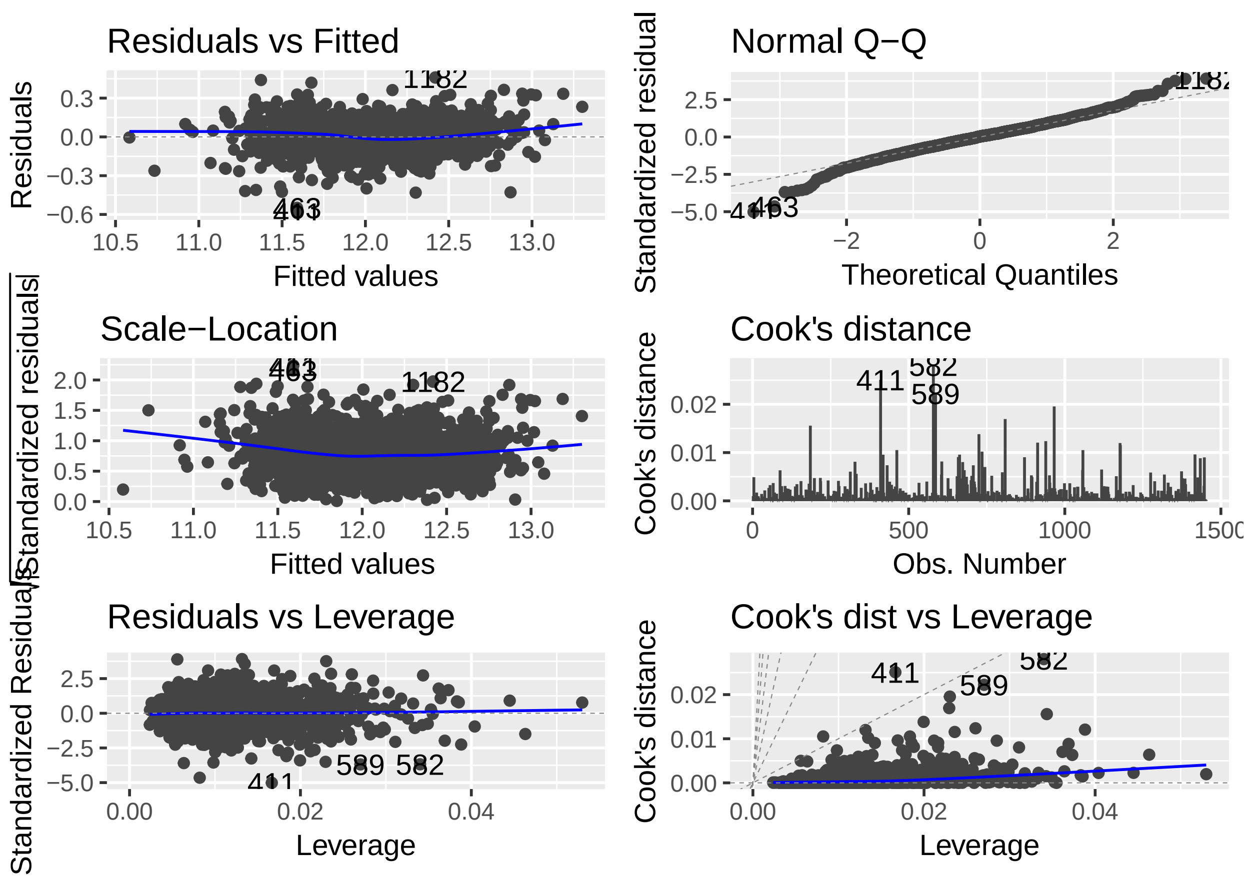
```
form4 <- paste(selected_predictors1, collapse = ' + ')
form4 <- formula(paste0('logSalePrice ~ ', form4))
```

Previously we just include all data in training part of `train` set, here we removed some outliers based on diagonis

```
linReg_model4 <- lm(form4, data = train_na_filled[-c(31, 496, 633, 524,
                                                     969, 1299, 1325), ])
linReg_model4_smy <- summary(linReg_model4)
linReg_model4_smy$r.squared
```

```
## [1] 0.9105422
```

```
ggplot2::autoplot(linReg_model4, 1:6)
```




```
train_pred_linReg_model4 <- predict(linReg_model4,
                                    newdata = train_na_filled[label != 5, ])
mean((train_pred_linReg_model4 - train_na_filled[label != 5, ]$logSalePrice)^2) %>%
  sqrt()
```

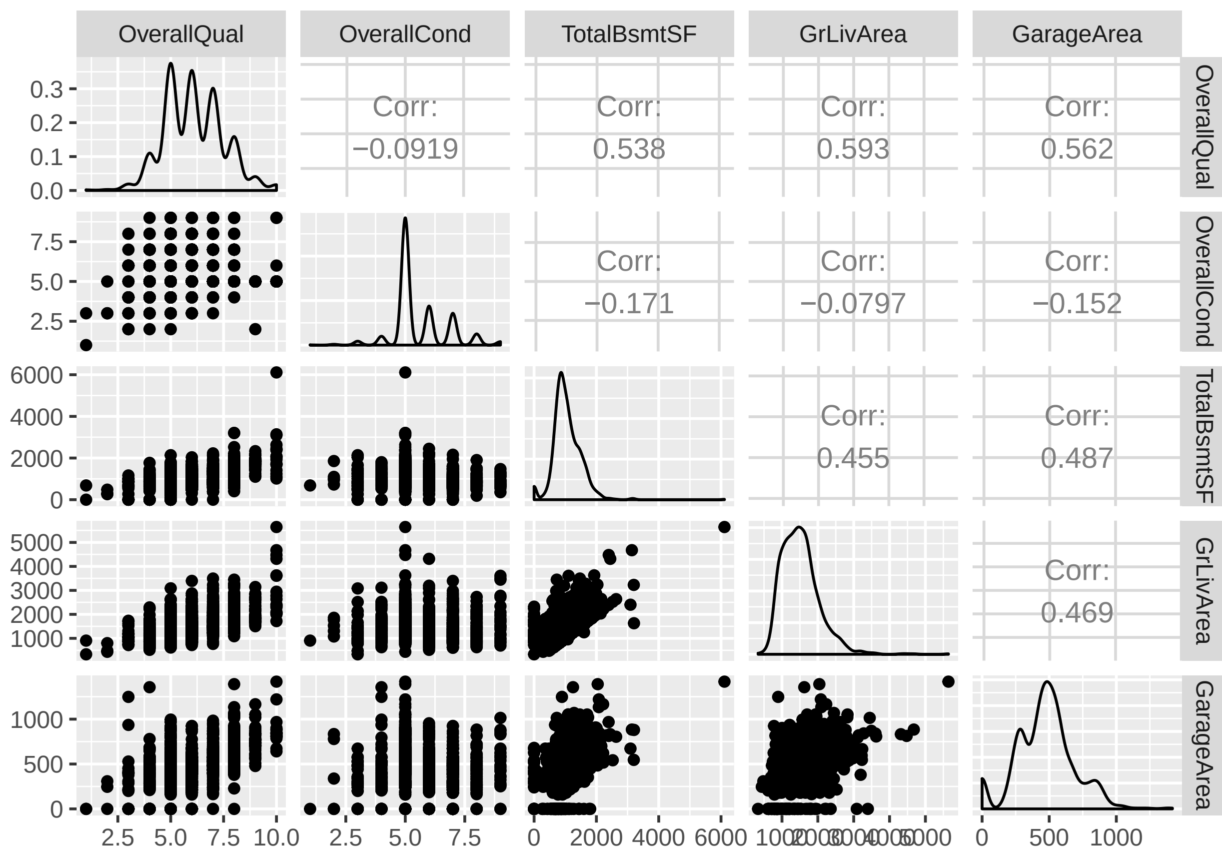
```
## [1] 0.1434153
```

```r
test_pred_linReg_model4 <- predict(linReg_model4,
                                   newdata = train_na_filled[label == 5, ])
mean((test_pred_linReg_model4 - train_na_filled[label == 5, ]$logSalePrice)^2) %>%
  sqrt()
```

```
## [1] 0.1115317
```

```r
# use best subset
# linReg_model6 <- regsubsets(form1,
#                             data = train_na_filled[label != 5, ],
#                             method = 'exhaustive', nvmax = 12)
# linReg_model6_smy <- summary(linReg_model6)
# linReg_model6_smy$rss
# ggplot2::autoplot(linReg_model1)
```

```r
selected_predictors1_copy <- selected_predictors1
selected_predictors1_copy[9] <- 'GrLivArea'
ggpairs(train_na_filled[, selected_predictors1_copy[c(3, 4, 6, 9, 15)]])
```




```r
form6 <- paste(selected_predictors1, collapse = ' + ')
form6 <- paste0(form6, '+OverallQual:OverallCond+TotalBsmtSF:GrLivArea:GarageArea')
form6 <- formula(paste0('logSalePrice ~ ', form6))
```

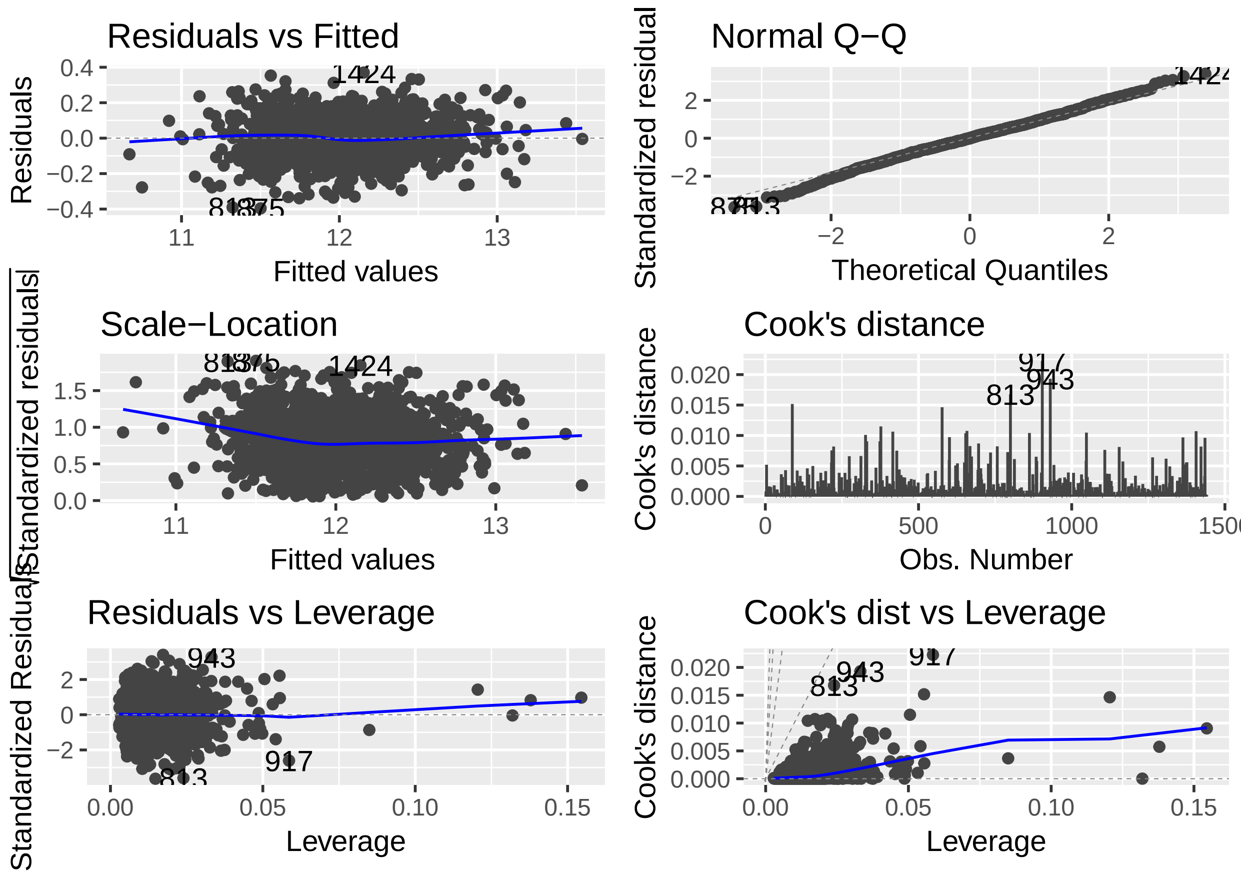
Previously we just include all data in training part of `train` set, here we removed some outliers based on diagonis

```r
linReg_model8 <- lm(form6, data = train_na_filled[-c(1299, 524, 1325, 633,
                                                     496, 31, 969, 582, 1182,
```

```
                                                      411, 711, 186, 1433, 689,
                                                      463, 729, 971, 739, 589), ])
linReg_model8_smy <- summary(linReg_model8)
linReg_model8_smy$r.squared
```

```
## [1] 0.9215063
```

```
ggplot2::autoplot(linReg_model8, 1:6)
```




```
train_pred_linReg_model8 <- predict(linReg_model8,
                                    newdata = train_na_filled[-c(1299, 524,
                                                                 1325, 633,
                                                                 496, 31,
                                                                 969, 582, 1182), ])
mean((train_pred_linReg_model8 - train_na_filled[-c(1299, 524,
                                                    1325, 633,
                                                    496, 31, 969,
                                                    582, 1182), ]$logSalePrice)^2) %>%
  sqrt()
```

```
## [1] 0.1154518
```

```
test_pred_linReg_model8 <- predict(linReg_model8,
                                   newdata = train_na_filled[label == 5, ])
mean((test_pred_linReg_model8 - train_na_filled[label == 5, ]$logSalePrice)^2) %>%
  sqrt()
```

```
## [1] 0.1098401
```

### (2) Ridge Regression

```
selected_predictors2 <- selected_predictors1
selected_predictors2[9] <- 'GrLivArea'
X2 <- as.matrix(train_na_filled[colnames(train_na_filled)%in%selected_predictors2])
# logarithm transformation
X2[, 9] <- log(X2[, 9])
interac_3n4 <- X2[, 3] * X2[, 4]
interac_6n9n15 <- X2[, 6] * X2[, 9] * X2[, 15]
X2 <- cbind(X2, interac_3n4, interac_6n9n15)
y2 <- as.matrix(train_na_filled['logSalePrice'])
```

```
lambda <- 10^seq(10, -2, length = 100)
```

```
set.seed(100)
ridge_model2 <- glmnet(X2[-c(1299, 524, 1325, 633,
                             496, 31, 969, 582, 1182,
                             411, 711, 186, 1433, 689,
                             463, 729, 971, 739, 589), ],
                       y2[-c(1299, 524, 1325, 633,
                             496, 31, 969, 582, 1182,
                             411, 711, 186, 1433, 689,
                             463, 729, 971, 739, 589)],
                       alpha = 0, lambda = lambda)
ridge_cv_out2 <- cv.glmnet(X, y, alpha = 0)
best_lambda <- ridge_cv_out2$lambda.min
```

```
train_pred_ridge_model2 <- predict(ridge_model2, s = best_lambda,
                                   newx = X2[-c(1299, 524, 1325, 633,
                             496, 31, 969, 582, 1182,
                             411, 711, 186, 1433, 689,
                             463, 729, 971, 739, 589), ])
mean((train_pred_ridge_model2 - y2[-c(1299, 524, 1325, 633,
                             496, 31, 969, 582, 1182,
                             411, 711, 186, 1433, 689,
                             463, 729, 971, 739, 589)])^2) %>% sqrt()
```

```
## [1] 0.1131399
```

```
test_pred_ridge_model2 <- predict(ridge_model2, s = best_lambda,
                                  newx = X2[label == 5, ])
mean((test_pred_ridge_model2 - y2[label == 5])^2) %>% sqrt()
```

```
## [1] 0.1143967
```

## 5. Prediction

```
convertFactor <- function(.variable) {
  if (class(test_na_filled[, .variable]) == 'character') {
    test_na_filled[, .variable] <-
      factor(test_na_filled[, .variable],
             levels = levels(train_na_filled[, .variable]))
  } else {
    test_na_filled[, .variable] <- test_na_filled[, .variable]
```

```
  }
}
```

```
load("C:/Users/Bangda/Desktop/kaggle/housing-price/eda.RData")
```

```
for (i in 1:ncol(train_na_filled)) {
  train_na_filled[, i] <- reArrangeOrder(colnames(train_na_filled)[i])
}
```

```
for (i in 1:ncol(test_na_filled)) {
  if (class(test_na_filled[, i]) == 'character') {
    test_na_filled[, i] <-
      factor(test_na_filled[, i],
             levels = levels(train_na_filled[, i])) %>% as.numeric()
  } else {
    test_na_filled[, i] <- test_na_filled[, i]
  }
}
```

```
X_pred <- as.matrix(test_na_filled[colnames(test_na_filled)%in%selected_predictors2])
X_pred[, 9] <- log(X_pred[, 9])
interac_3n4_pred <- X_pred[, 3] * X_pred[, 4]
interac_6n9n15_pred <- X_pred[, 6] * X_pred[, 9] * X_pred[, 15]
X_pred <- cbind(X_pred, interac_3n4_pred, interac_6n9n15_pred)
```

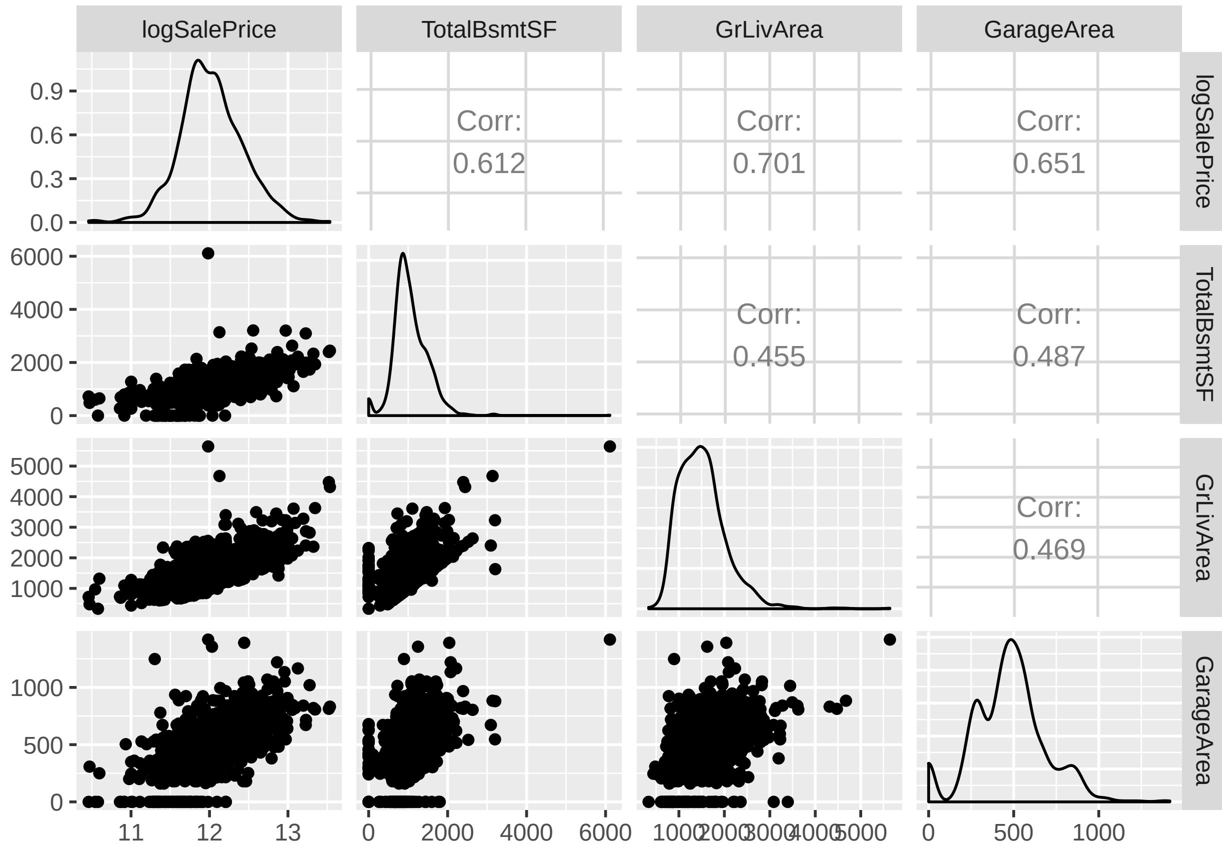
```
pred_logSalePrice <- predict(ridge_model2, s = best_lambda,
                             newx = X_pred)
test$SalePrice <- exp(pred_logSalePrice)
setwd('C:/Users/Bangda/Desktop/kaggle/housing-price')
write.csv(test[, c('Id', 'SalePrice')], 'prediction1.csv', row.names = FALSE)
```

```
pred_logSalePrice <- predict(linReg_model4, newdata = test_na_filled)
test$SalePrice <- exp(pred_logSalePrice)
write.csv(test[, c('Id', 'SalePrice')], 'prediction2.csv', row.names = FALSE)
```

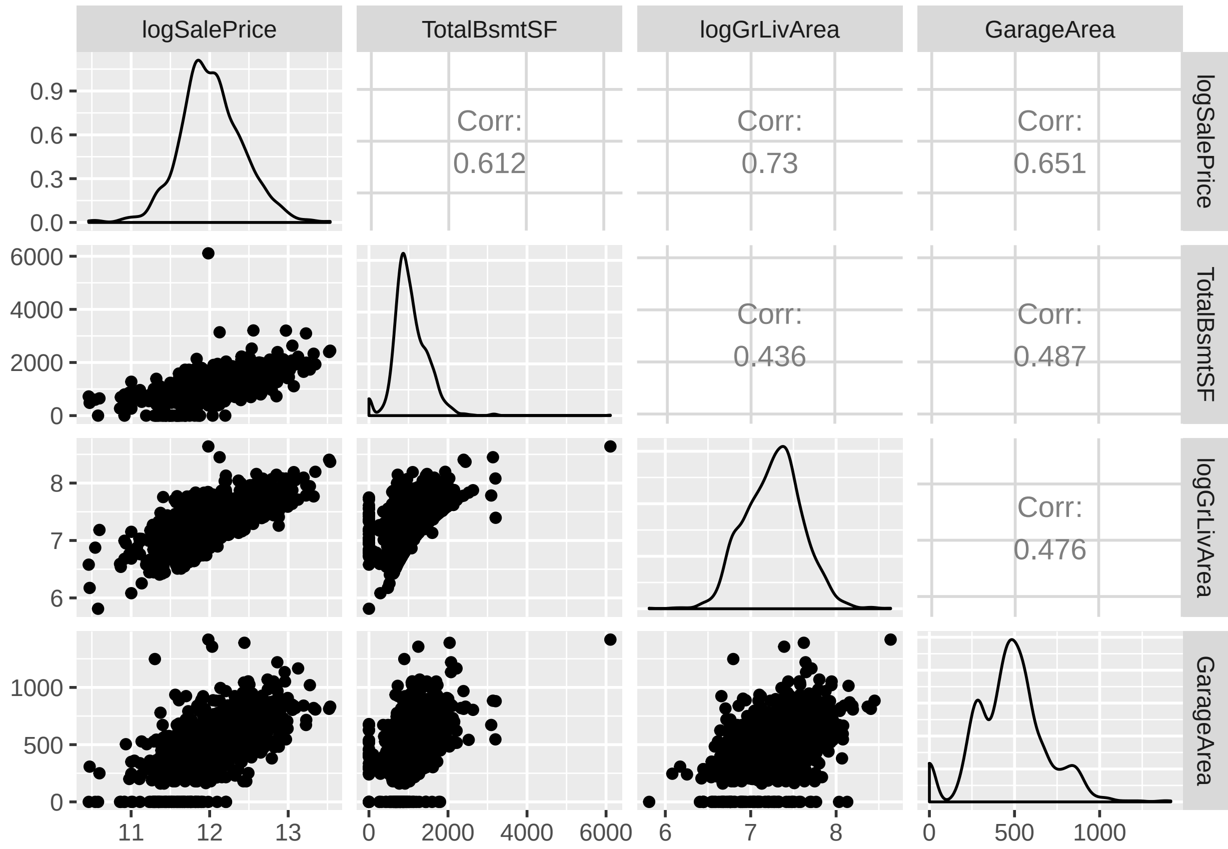
```
linReg_model5 <-
  lm(logSalePrice ~ MSZoning + factor(Neighborhood) +
       OverallQual + OverallCond + BsmtQual +
       TotalBsmtSF + HeatingQC + CentralAir +
       GrLivArea + KitchenQual + Fireplaces +
       GarageFinish + GarageCars + GarageArea +
       SaleCondition + factor(MoSold),
     data = train_na_filled)
linReg_model5_smy <- summary(linReg_model5)
```

```
pred_logSalePrice <- predict(linReg_model5, newdata = test_na_filled_copy)
test$SalePrice <- exp(pred_logSalePrice)
write.csv(test[, c('Id', 'SalePrice')], 'prediction3.csv', row.names = FALSE)
```

```
train_na_filled %>%
  select(logSalePrice, TotalBsmtSF, GrLivArea, GarageArea) %>%
  ggpairs()
```



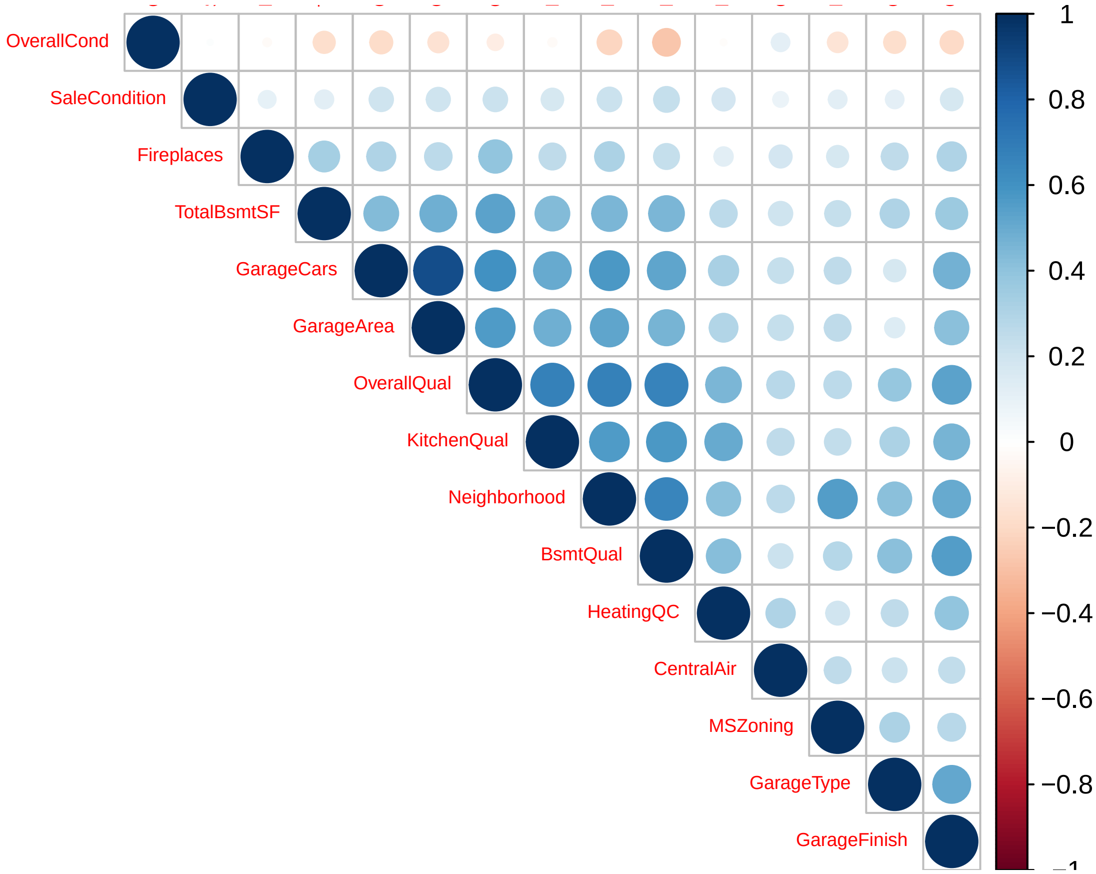

```
train_na_filled %>%
  mutate(logGrLivArea = log(GrLivArea)) %>%
  select(logSalePrice, TotalBsmtSF, logGrLivArea, GarageArea) %>%
  ggpairs()
```



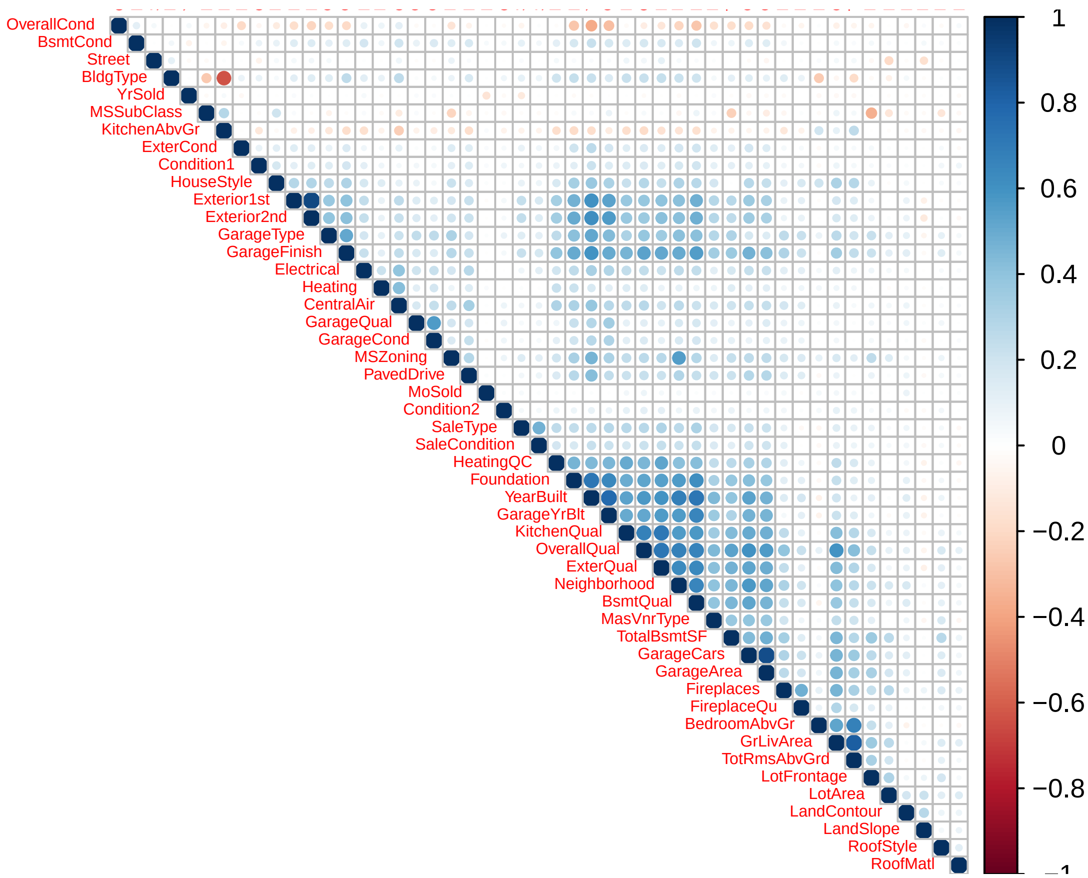

```
pred_logSalePrice <- predict(linReg_model8, newdata = test_na_filled)
test$SalePrice <- exp(pred_logSalePrice)
write.csv(test[, c('Id', 'SalePrice')], 'prediction5.csv', row.names = FALSE)
```

### 6. Principal Component Regression

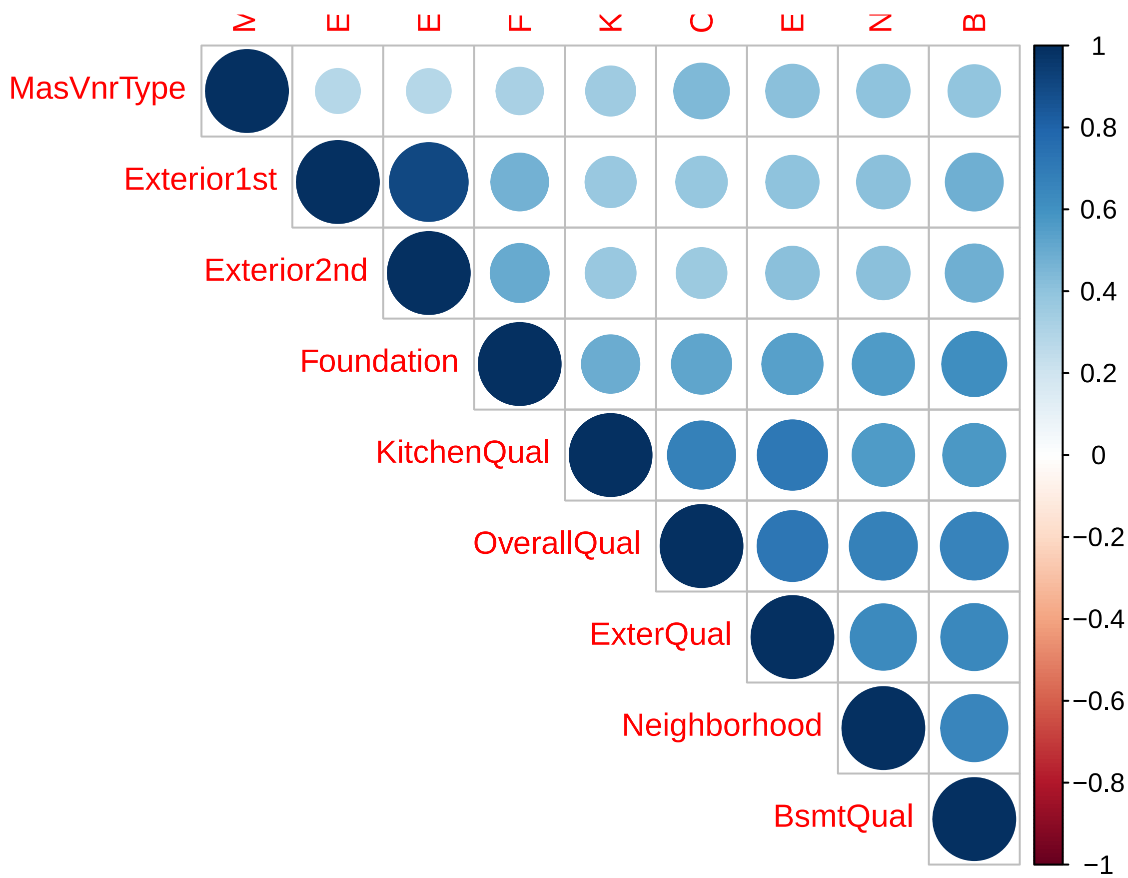
```
for (i in 1:ncol(train_na_filled)) {
  train_na_filled[, i] <- as.numeric(reArrangeOrder(colnames(train_na_filled)[i]))
}
corMat <- cor(train_na_filled[, colnames(train_na_filled)%in%selected_predictors1])
corrplot(corMat, type = 'upper', order = 'hclust', tl.cex = .6)
```




```
corMat2 <- cor(train_na_filled[, colnames(train_na_filled)%in%predictors])
corrplot(corMat2, type = 'upper', order = 'hclust', tl.cex = .5)
```




```
high_correlate <- c(
  'Foundation', 'KitchenQual',  'OverallQual',
  'ExterQual',  'Neighborhood', 'BsmtQual',
  'MasVnrType', 'Exterior1st',  'Exterior2nd'
)
corMat3 <- cor(train_na_filled[, colnames(train_na_filled)%in%high_correlate])
corrplot(corMat3, type = 'upper', order = 'hclust')
```

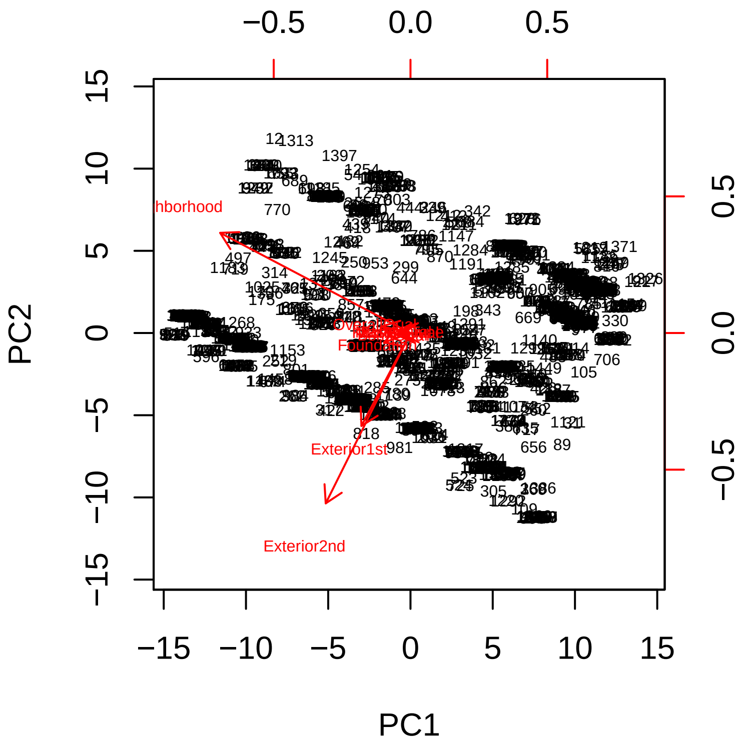



```
X_4pca <- X[, colnames(X)%in%high_correlate]
```

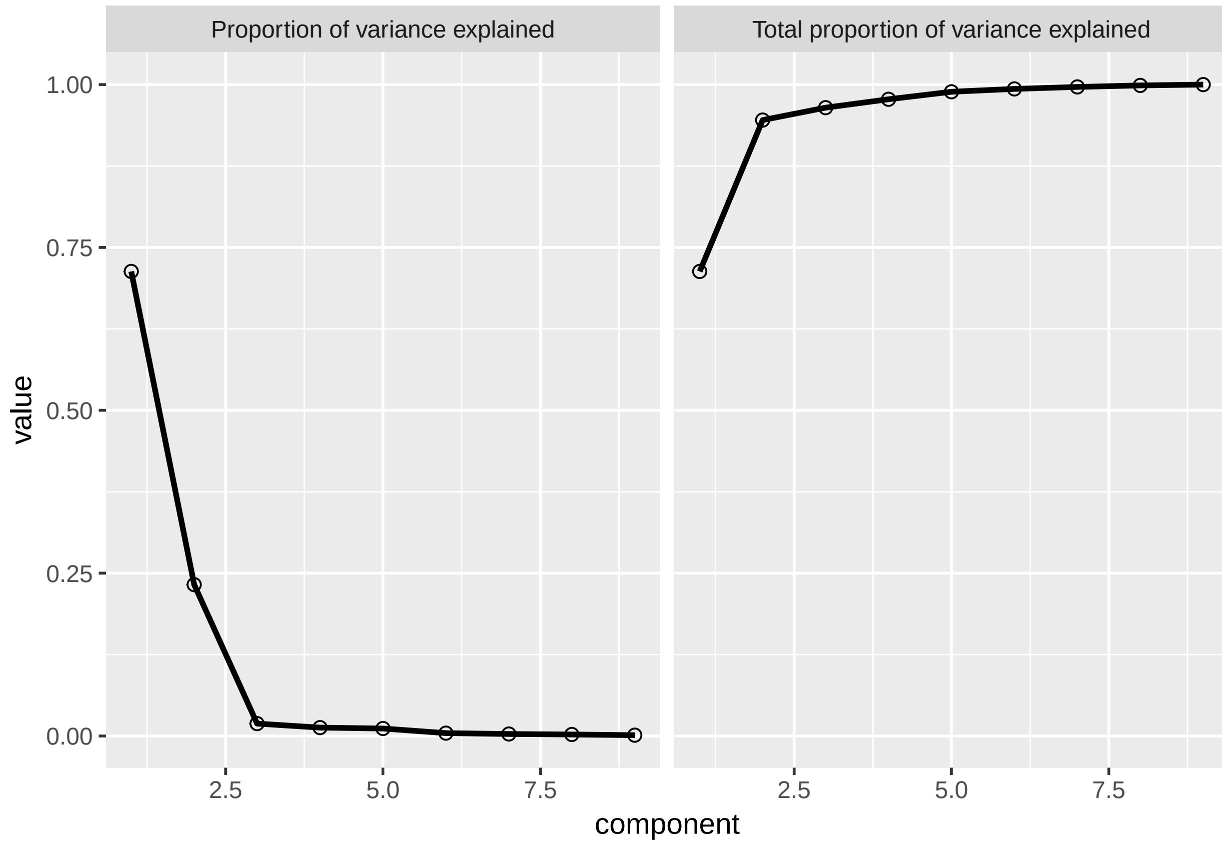
```
pr_out <- prcomp(X_4pca)
pr_out$rotation
```

```
##                    PC1          PC2          PC3         PC4         PC5
## Neighborhood -0.86982936  0.458073998  0.178626968 -0.02080902  0.02780606
## OverallQual  -0.13030711  0.030414875 -0.633998721  0.32290500 -0.53070441
## Exterior1st  -0.22413209 -0.424824129  0.007449431  0.73497966  0.47819937
## Exterior2nd  -0.38728431 -0.778976531  0.112469380 -0.37531378 -0.29896046
## MasVnrType   -0.03991824 -0.001434736 -0.129346168  0.06921627 -0.10286782
## ExterQual    -0.05249740  0.003318788 -0.194588553  0.05271160 -0.10912617
## Foundation   -0.12090170 -0.043474712 -0.637194378 -0.44776023  0.59925801
## BsmtQual     -0.06534171 -0.007439296 -0.200354369  0.01630771 -0.02981059
## KitchenQual  -0.05437293  0.002168824 -0.229520127  0.07693665 -0.13053169
##                     PC6          PC7          PC8          PC9
## Neighborhood -0.008370112  0.011276768 -0.015349539  0.004298852
## OverallQual  -0.194225084  0.382572130 -0.081826019  0.058698319
## Exterior1st  -0.011227472  0.005673725 -0.012116012 -0.007681108
## Exterior2nd  -0.008242044  0.011435803 -0.009939155  0.004667370
## MasVnrType    0.979704843  0.052488727 -0.053692106  0.029015081
## ExterQual     0.004430202 -0.398986763 -0.003404063 -0.886261597
## Foundation   -0.004958270  0.091313020 -0.097371682  0.005724483
## BsmtQual      0.025439976 -0.217201193  0.940751802  0.146768961
## KitchenQual  -0.038724498 -0.797426173 -0.308923424  0.434257817
```

```r
biplot(pr_out, scale = 0, cex = .5)
```




```r
pr_var <- (pr_out$sdev)^2
pr_pve <- pr_var / sum(pr_var)
pr_cpve <- cumsum(pr_pve)
pr_pve_df <- data.frame(
  component = 1:9,
  prop_var_exp = pr_pve,
  cumu_prop_var_exp = pr_cpve
)
pr_pve_df %>%
  melt(id.vars = 'component') %>%
  ggplot() +
  geom_line(aes(x = component, y = value), size = 1) +
  geom_point(aes(x = component, y = value), size = 2, shape = 1) +
  facet_wrap(~ variable, labeller = as_labeller(
    c('prop_var_exp' = 'Proportion of variance explained',
      'cumu_prop_var_exp' = 'Total proportion of variance explained')))
```




```r
ldMat <- pr_out$rotation[, 1:2]
pca_score <- X_4pca %*% ldMat
colnames(pca_score) <- c('PC1', 'PC2')
selected_predictors3 <- setdiff(selected_predictors1, high_correlate)
train_na_filled$PC1 <- pca_score[, 1]
train_na_filled$PC2 <- pca_score[, 2]
selected_predictors3 <- c(selected_predictors3, 'PC1', 'PC2')
```

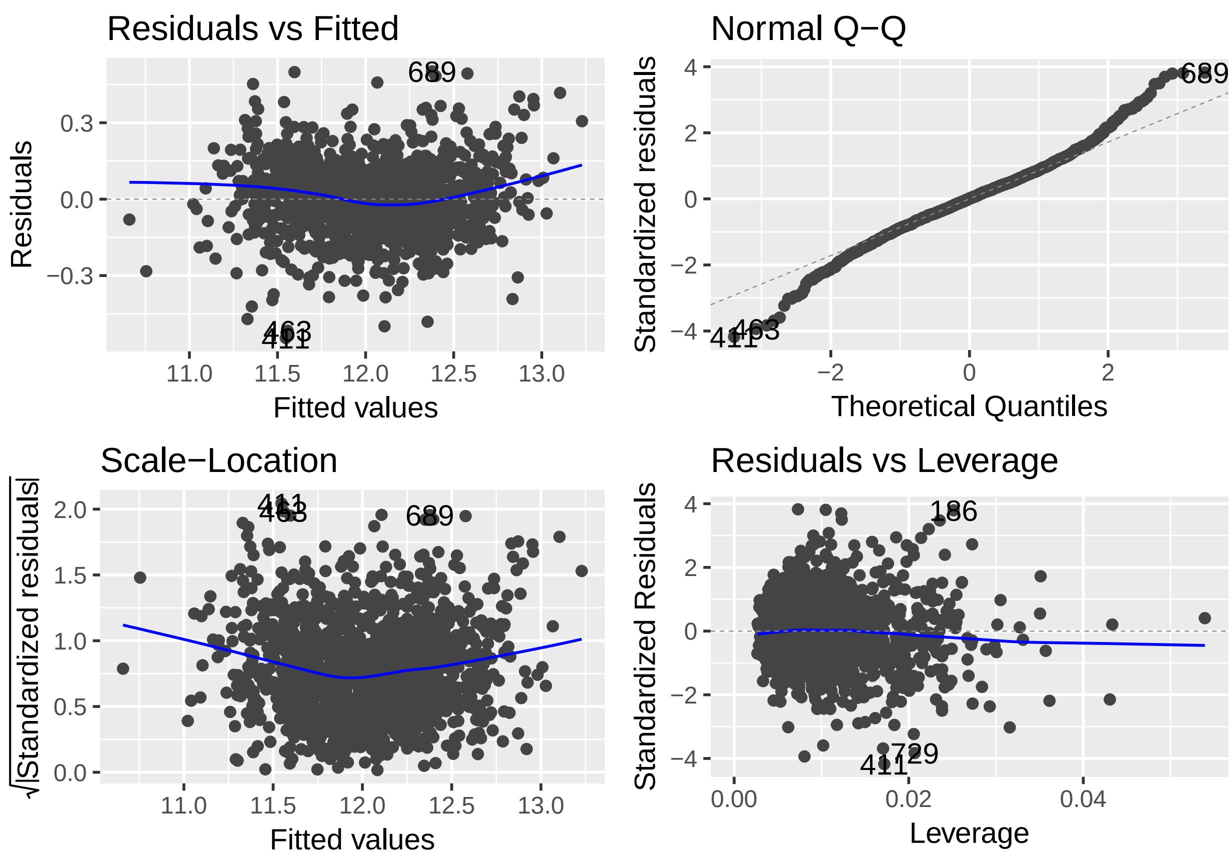
```r
form5 <- paste(selected_predictors3, collapse = ' + ')
form5 <- formula(paste0('logSalePrice ~ ', form5))
```

Previously we just include all data in training part of `train` set, here we removed some outliers based on diagonis

```r
linReg_model7 <- lm(form5, data = train_na_filled[-c(633, 524, 1299,
                                                     1325, 969, 31, 496), ])
linReg_model7_smy <- summary(linReg_model7)
linReg_model7_smy$r.squared
```

```
## [1] 0.8894852
```

```r
ggplot2::autoplot(linReg_model7)
```




```
train_pred_linReg_model7 <- predict(linReg_model7,
                                    newdata = train_na_filled[-c(633, 524, 1299,
                                                                 1325, 969, 31, 496), ])
mean((
  train_pred_linReg_model7 -
    train_na_filled[-c(633, 524, 1299, 1325, 969, 31, 496), ]$logSalePrice)^2) %>%
  sqrt()
```

```
## [1] 0.1309873
```

```
X_pred2 <- as.matrix(test_na_filled[colnames(test_na_filled)%in%predictors])
X_pred2[, 9] <- log(X_pred2[, 9])
X_4pca_pred <- X_pred2[, colnames(X)%in%high_correlate]
ldMat_pred <- pr_out$rotation[, 1:2]
pca_score_pred <- X_4pca_pred %*% ldMat_pred
test_na_filled$PC1 <- pca_score_pred[, 1]
test_na_filled$PC2 <- pca_score_pred[, 2]
pred_logSalePrice <- predict(linReg_model7, newdata = test_na_filled)
test$SalePrice <- exp(pred_logSalePrice)
write.csv(test[, c('Id', 'SalePrice')], 'prediction4.csv', row.names = FALSE)
```

**linReg__model8 is best for now.**