Predicting House Price in King County

Xinlei Chen, Xinyi Lin, Jiawei Ye

BACKGROUND

The aim of this project is to use the house price data of King County, the United States, obtained from kaggle (https://www.kaggle.com/harlfoxem/housesalesprediction) to do the house price perdiction. The original dataset includes houses sold between May 2014 and May 2015 and it has 21613 observations and 21 variables. We built 4 linear models (ridge, LASSO, PCR and stepwise regression) and 2 non-linear models (GAM, MARS) to do the prediction, and used cross-validation methods for model comparison.

EXPLORATORY DATA ANALYSIS

The original dataset contains 21613 observations and 21 variables.

The Response Variable: Price

The left-hand side of *Fig.1* is the distribution of price which is the response variable. As we can see the distribution is extremly right-skewed, as few people can afford very expensive houses. To fix this we took the log of price. The right-hand side of *Fig.1* is the distribution of price after log-transformation.

Correlation and Collinearity

We checked the correlation within predictors as well as between predictors and response (*Fig.2*). If the correlation between perdictors equals to 0 it means these together won't add anything to the model and we should get rid of such collinear variables. In this data, the highest multicollinearity (0.88) comes from the square footage of house apart from basement (sqft_above) and the square footage of the home (sqft_living), and the second highest multicollinearity (0.76) comes from the square footage of the home (sqft_living) and the living room area in 2015 (sqft_living15).

Variables with Near-Zero Variance

Near-zero variance predictors is constant or almost constant predictors across samples. This kind of predictors is non-informative and it can break some models we may want to fit to our data. We checked for near-zero variance predictors and the result showed that there were 4 predictors had near-zero variance (waterfront, view, sqft_basement and yr_renovated). So, we removed them all.

METHODS

After initial variable selection, we used all 13 variables (25 predictors) and **caret** package to fit all 6 models.

Linear Method

1. Ridge Regression

Ridge regression is a coefficient shrinkage method. It's similar to least squares but the coefficients are estimated with a penalty term $\lambda \sum_j \beta_j^2$. When $\lambda = 0$ ridge regression is the same as least squares; as $\lambda \to \infty$ the impact of the shinkage grows. Ridge regression shrinkes estimated coefficients of all variables. The best lambda of our ridge model is about 0.036.

2. The LASSO

The LASSO regression is also a coefficient shrinkage method. The differences between ridge regression and the LASSO regression is that the LASSO performs variable selection, it can shrink coefficient to zero and not necessary include all predictors in the final model. The LASSO regression estimates coefficients with penalty term $\lambda \sum_j |\beta_j|$. In our final LASSO model, the best lambda is 0.0004 and it includes 23 perdictors. The LASSO only shrinkes 2 coefficients(`condition3` and `grade8`) to zero.

3. Principal Component Regression

PCR is a dimension reduction method, it involves constructing the first M pricipal components and then using these components as the predictors in a linear regression model that is fit using least squares. The final model of PCR has 14 components (*Fig.5*).

4. Stepwise Regression

Stepwise regression model was fitted. Stepwise regression selects the best subset. The adjusted R-squared is 0.756. The fitted model is: price ~ bedrooms + bathrooms + sqft_living + sqft_lot + floors + condition2 + condition3 + condition4 + condition5 + grade4 + grade6 + grade7 + grade8 + grade9 + grade10 + grade11 + grade12 + grade13 + sqft_above + yr_built + lat + long + sqft_living15

Non-Linear Method

1. Generalized Additive Model

Generalized additive models (GAMs) provide a general framework for extending a standard linear model by allowing non-linear functions of each ofthevariables,while maintaining additivity. A total of 19 predictors were included in the GAM model. Numerical feature total number of floors in the house and cateogrical features overall condition and grading of the house are chosen as linear predictors. The number of bedrooms and bathrooms, built year, square footage of living room area, the longitude and latitude coordinates, square footage of house apart from basement and lot size were selected as non linear predictors.

2. Multivariate Adaptive Regression Spline

Multivariate Adaptive Regression Spline is a non-parametric regression technique and can be seen as an extension of linear models that automatically models nonlinearities and interactions between variables.

In order to avoid overfitting, for the MARS method we manually set the upper limit of number of variables to be 10 and only two way interaction terms can be included in the data. The square footage of living area, longitude and latitude coordinates, interaction terms between longitudine and latitude coordinates, interaction terms between longitude coordinates and square footage of living area in 2015, interaction terms between latitude coordinates and square footage of living area in 2015 were included in the model.

Generally speaking, the house price increases as the living area increase when the living area is greater than a cut off point while the price decreases as the living area increase when the living area is smaller than the cut off point.

RESULT: MODEL COMPARISON

	Train RMSE	Test RMSE
lasso	0.2608836	0.2628065
ridge	0.2620460	0.2612972

	Train RMSE	Test RMSE
pcr	0.2769255	0.2784702
stepwise	0.2608783	0.2784702
GAM	0.2151794	0.2185575
MARS	0.2315682	0.2330594

According to the linear model comparison result, GAM is the best model with lowest training RMSE 0.215 and testing RMSE 0.219. MARS is the second best model with training RMSE 0.232 and testing RMSE 0.233. Non-linear models overall perform better than linear models. Among 4 linear models, the LASSO model is slightly better than the stepwise linear model and it's the best linear model. The training and testing RMSE are close to each other among 4 linear models, and also among 2 non-linear models.

DISCUSSION

Comparing the training RMSE of all 6 models, non-linear models perform better than linear-models and GAM is the best model. Among the linear methods, ridge and the LASSO regression models are not suitable since they do not shink many coefficients toward 0. So for prediction purpose, we suggest GAM and MARS.

APPENDIX

Figures

Fig.1: the distribution of response variable

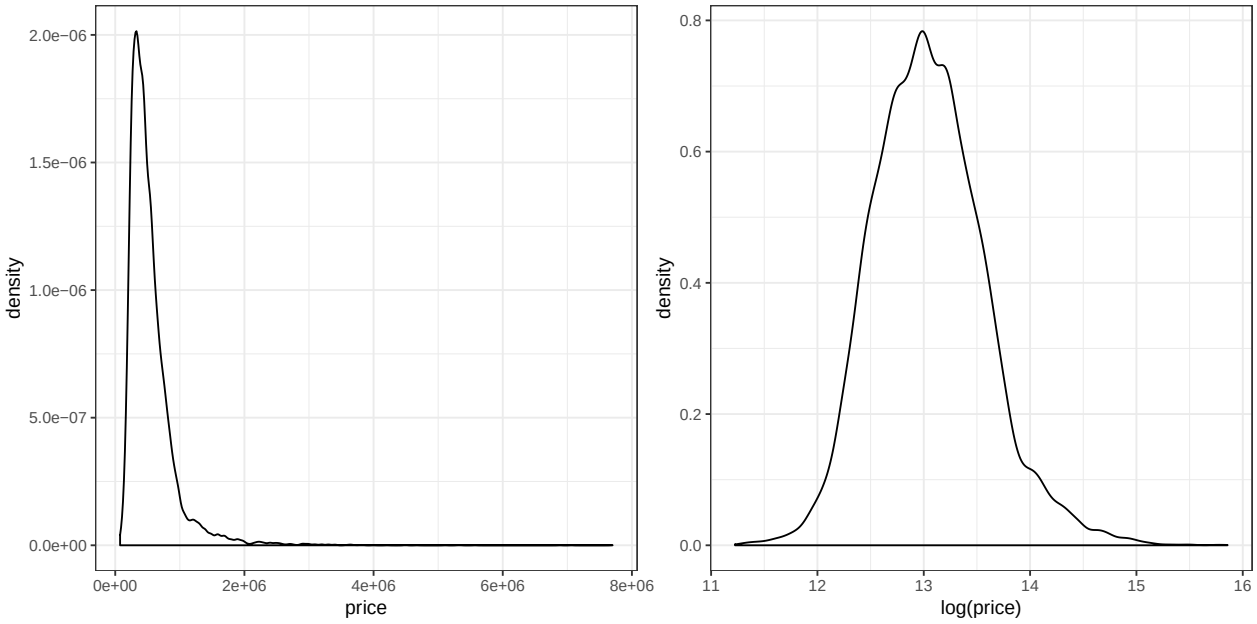


Fig.2: correlation plot

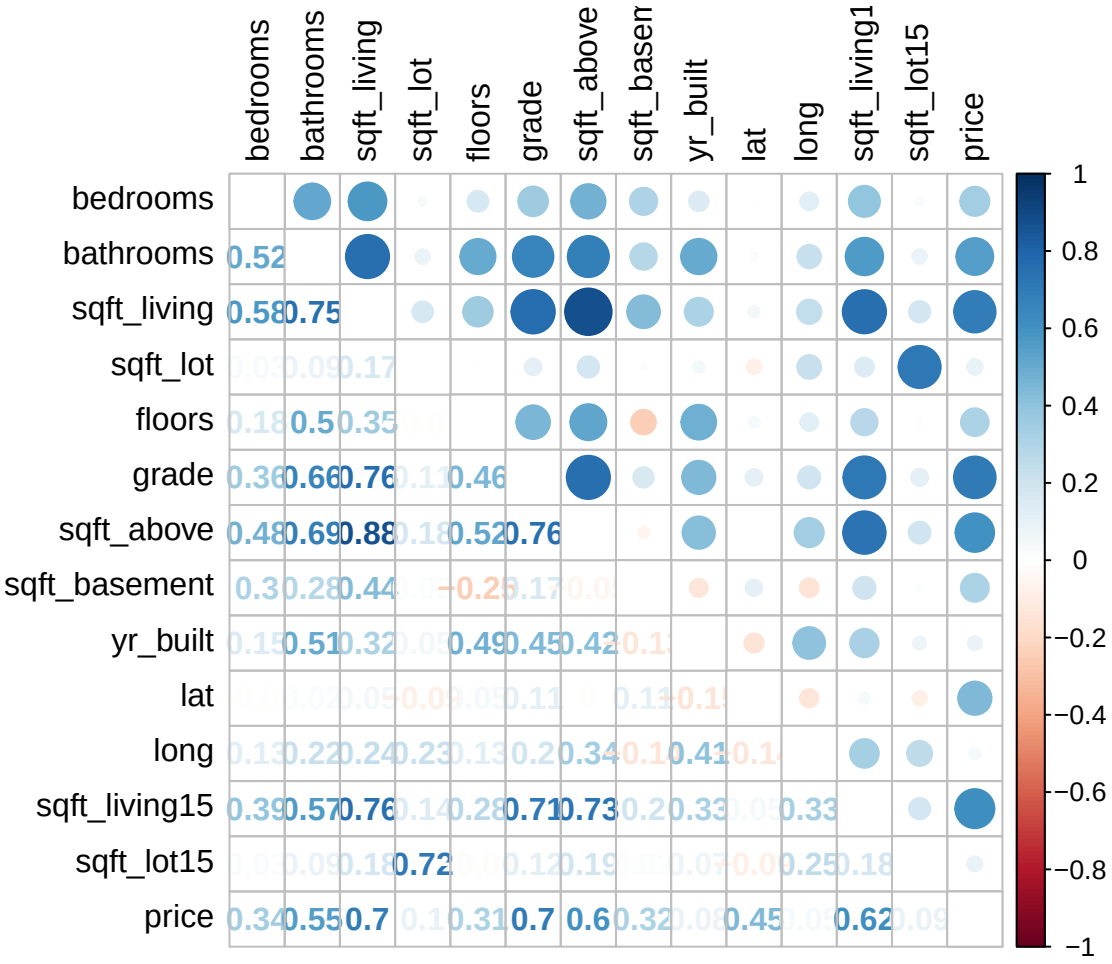


Fig.3: ridge regression

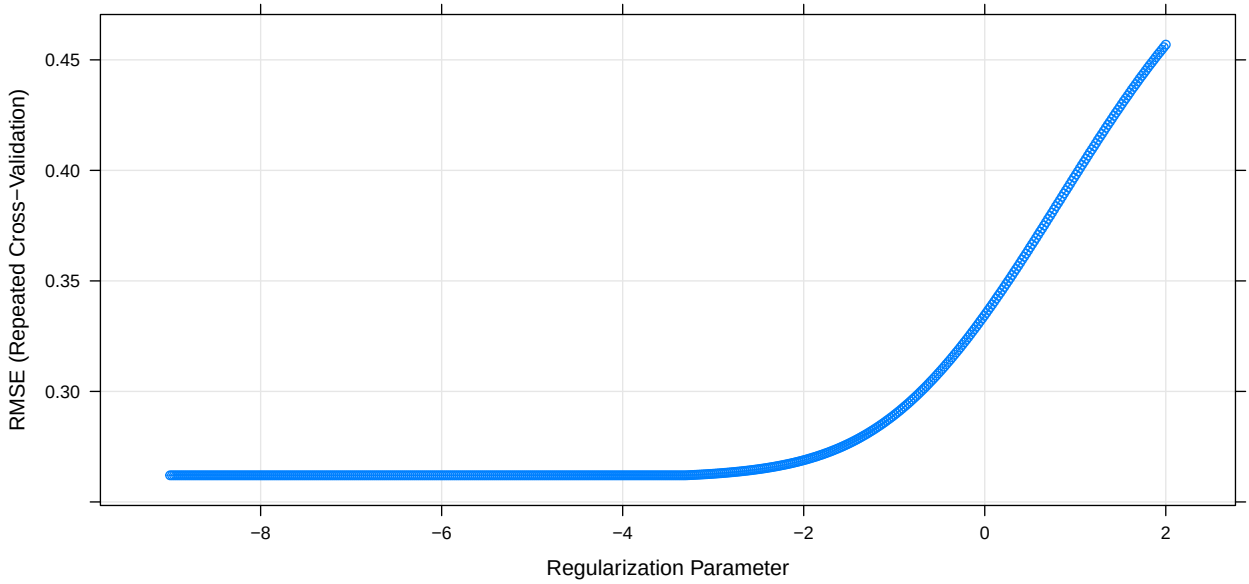


Fig.4: the LASSO regression

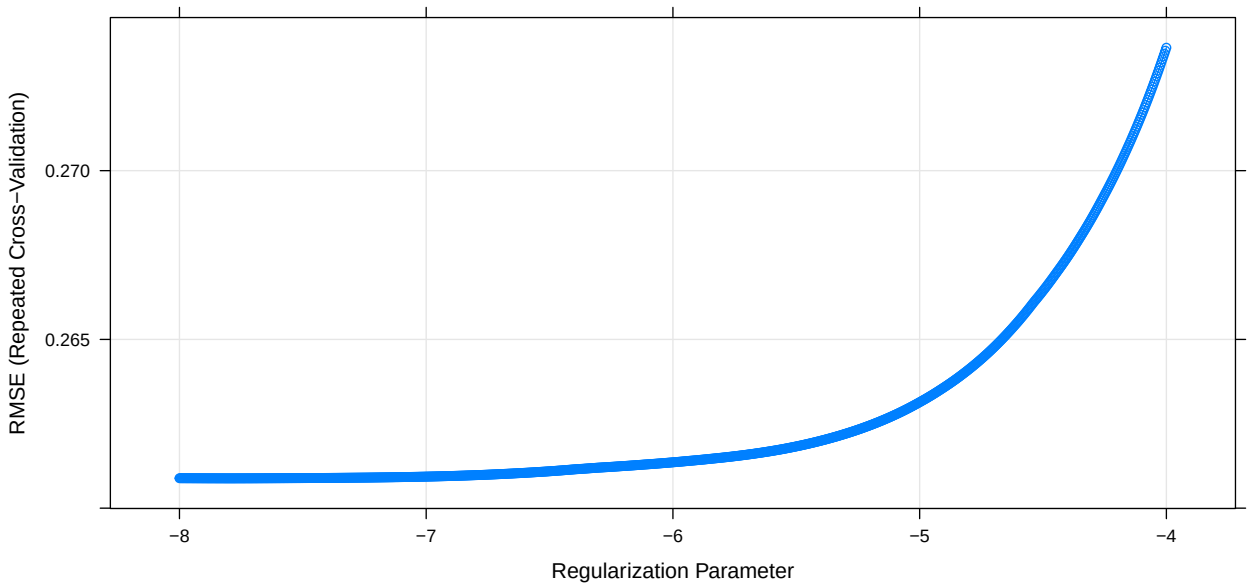


Fig.5: PCR

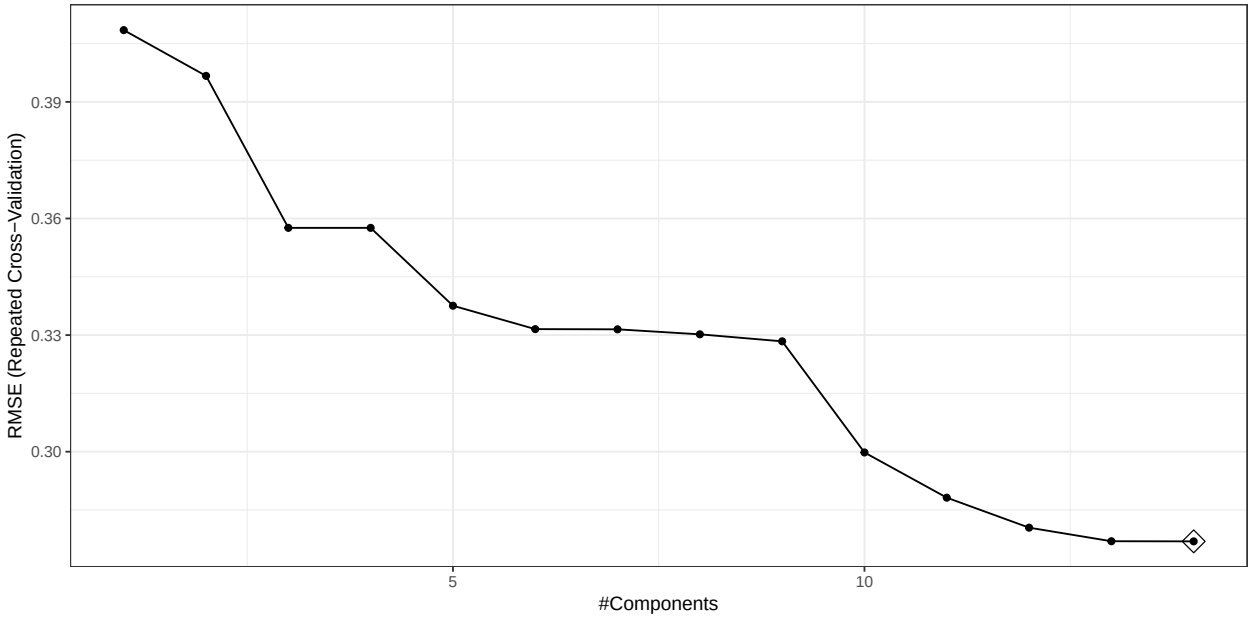
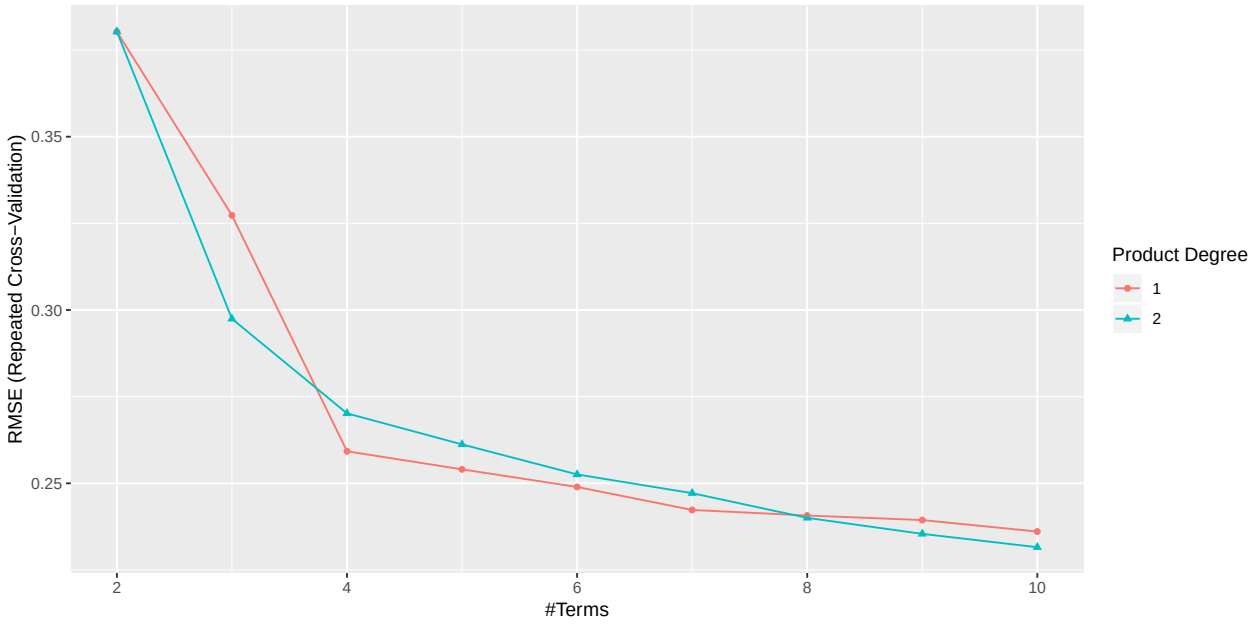


Fig.6: MARS



Code

```
library(tidyverse)
library(glmnet)
library(caret)
library(patchwork)
# import data and change some variables into factor
# we don't change yr_built into factor because usually house price increase with years
data = read.csv("./data/kc_house_data.csv") %>%
  mutate(waterfront = as.factor(waterfront),
         view = as.factor(view),
         condition = as.factor(condition),
         yr_renovated = as.factor(yr_renovated))

new_data = data %>%
  janitor::clean_names() %>%
  select(-id, -date, -zipcode) %>%
  select(-price, price) %>%
  mutate(price = log(price))
p1 = data %>%
  ggplot(aes(x = price)) +
  geom_density() +
  theme_bw()

p2 = new_data %>%
  ggplot(aes(x = price)) +
  geom_density() +
  theme_bw() +
  labs(
    x = 'log(price)'
  )
```

```
p3=p1+p2
numericVars <- which(sapply(new_data, is.numeric)) #index vector numeric variables
train_numVar <- new_data[, numericVars]
cor_numVar <- cor(train_numVar, use="pairwise.complete.obs") #correlations of all numeric variables
# delete variables with near_zero variance
#str(new_data)
#nearZeroVar(new_data)
near_zero_var = c(6,7,11,13)
data = new_data[, -near_zero_var]
# delete rows containing the missing data
data = na.omit(data)

# split train and test data
set.seed(123)
smp_size <- floor(0.5 * nrow(data))
train_ind <- sample(seq_len(nrow(data)), size = smp_size)

train <- data[train_ind, ]
test <- data[-train_ind, ]

# write train and test file
write_csv(train, path = "data/house_train.csv")
write_csv(test, path = "data/house_test.csv")

train = read.csv("./data/house_train.csv") %>%
  mutate(condition = as.factor(condition),
         grade = as.factor(grade))
test = read.csv("./data/house_test.csv") %>%
  mutate(condition = as.factor(condition),
         grade = as.factor(grade))
x_train = model.matrix(price~.,train)[,-1]
y_train = train$price

x_test = model.matrix(price~.,test)[,-1]
y_test = test$price
# ridge regression
ctrl1 <- trainControl(method = "repeatedcv", number = 10, repeats = 5)
set.seed(123)
ridge.fit = train(x_train, y_train,
                  method = "glmnet",
                  tuneGrid = expand.grid(alpha = 0,
                                         lambda = exp(seq(-9, 2, length=500))),
                  trControl = ctrl1)


ridge.plot = plot(ridge.fit, xTrans = function(x) log(x))
best.lambda = ridge.fit$bestTune$lambda

ridge.full <- train(x_train, y_train,
                    method = 'glmnet',
                    trControl = trainControl(method = 'none'),
                    tuneGrid = expand.grid(
                      lambda = ridge.fit$bestTune$lambda, alpha = 0)
```

```
                    )

#coef(ridge.full$finalModel, s = ridge.fit$bestTune$lambda)

#mse
ridge.pred = predict(ridge.fit, s = best.lambda, newdata = x_test)
#lasso
set.seed(123)
lasso.fit <- train(x_train, y_train,
                   method = "glmnet",
                   tuneGrid = expand.grid(alpha = 1,
                                          lambda = exp(seq(-8, -4, length=1000))),
                   trControl = ctrl1)

lasso.plot = plot(lasso.fit, xTrans = function(x) log(x))
coef = coef(lasso.fit$finalModel,lasso.fit$bestTune$lambda)
#length(coef[coef != 0])

best.lambda = lasso.fit$bestTune

# mse
lasso.pred = predict(lasso.fit, s = best.lambda, newdata = x_test)

set.seed(123)
# pcr
ctrl1 <- trainControl(method = "repeatedcv", number = 10, repeats = 5)

pcr.fit <- train(x_train, y_train,
                 method = "pcr",
                 tuneLength = length(train),
                 trControl = ctrl1,
                 preProc = "scale")
ncomp = pcr.fit$bestTune$ncomp
pcr.plot = ggplot(pcr.fit, highlight = TRUE) + theme_bw()
# mse
pcr.pred = predict(pcr.fit, newdata = x_test)
lm.fit <- train(x_train, y_train,
                #method = "lm",
                method = 'glmStepAIC',
                trControl = ctrl1)
#summary(lm.fit)
lm.pred = predict(pcr.fit, newdata = x_test)
library(nlme)
library(mgcv)
library(pdp)
library(earth)

house_train = read.csv("./data/house_train.csv") %>%
              mutate(condition = as.factor(condition),
                     grade = as.factor(grade))
house_test = read.csv("./data/house_test.csv") %>%
             mutate(condition = as.factor(condition),
                    grade = as.factor(grade))
```

```r
ctrl1 <- trainControl(method = "repeatedcv", number = 10, repeats = 5)

x = model.matrix(price~., house_train)[,-1]
y = house_train$price

x_test = model.matrix(price~., house_test)[,-1]
y_test = house_test$price


#build model on train data
gam.fit <- train(x, y,
                 method = "gam",
                 tuneGrid = data.frame(method = "GCV.Cp", select = c(TRUE,FALSE)),
                 trControl = ctrl1)
#gam.fit$bestTune
#gam.fit$finalModel


#check predictive ability on test data
gam.pred = predict(gam.fit, newdata = x_test)
#build model on train data
mars_grid <- expand.grid(degree = 1:2,
                         nprune = 2:10)

set.seed(123)
mars.fit <- train(x, y,
                 method = "earth",
                 tuneGrid = mars_grid,
                 trControl = ctrl1)

mars.plot = ggplot(mars.fit)

#mars.fit$bestTune

#coef(mars.fit$finalModel)

#check predictive ability on test data
mars.pred = predict(mars.fit, newdata = x_test)
set.seed(123)
resamp <- resamples(list(lasso = lasso.fit, ridge = ridge.fit, pcr = pcr.fit, stepwise = lm.fit, GAM = g
data.frame(train_rmse = summary(resamp)$statistics$RMSE[,4],
           test_rmse = c(sqrt(mean((ridge.pred - y_test)^2)),
                         sqrt(mean((lasso.pred - y_test)^2)),
                         sqrt(mean((pcr.pred - y_test)^2)),
                         sqrt(mean((lm.pred - y_test)^2)),
                         sqrt(mean((gam.pred - y_test)^2)),
                         sqrt(mean((mars.pred - y_test)^2)))) %>%
             knitr::kable(col.names = c("Train RMSE", "Test RMSE"))
p3
corrplot::corrplot.mixed(cor_numVar, tl.col="black", tl.pos = "lt")
ridge.plot
lasso.plot
pcr.plot
```

```
mars.plot
```