Group-2

ThokalaVamsi & PenghuaWang

10/5/2022

Contents


```
#Install  missing package
list.of.packages <- c("jtools", "MASS", "dplyr", "flexmix", "regclass","ggResidpanel", "caret","DEoptimR")
new.packages <- list.of.packages[!(list.of.packages %in% installed.packages()[,"Package"])]
if(length(new.packages)) install.packages(new.packages)
library(jtools)
library(MASS)
library(flexmix)
library(ggResidpanel)
library(dplyr)
library(regclass)
library(caret)
```

Import packages

```
data <- read.csv("housingData.csv")
#drop ID variable
data$Id <- NULL
#create log of the price
data$SalePrice <- log(data$SalePrice)
#overview
#head(data)
```

Import data

Data Preparation

```r
data <- data[colSums(is.na(data))/nrow(data) < .3]
#new number of variables
paste("Number of remaining variables:",length(names(data)))
```

Drop variables with more than 30% missing observations

```
## [1] "Number of remaining variables: 68"
```

```r
#get character variables
cars <- unlist(lapply(data, is.character), use.names = FALSE)
#convert all character to factor
data <- as.data.frame(unclass(data),stringsAsFactors=TRUE)
#convert categorical to numeric
data[sapply(data, is.factor)] <- data.matrix(data[sapply(data, is.factor)])
#convert the numerical categorical variables to factor
data[,cars] <- lapply(data[,cars] , factor)


#Impute numerical with mean in numeric and mode in factor
#create function to compute the mode

var_mode <- function(x) {
    ij <- unique(x)
    ij[which.max(tabulate(match(x, ij)))]
}


data<- data %>% mutate_if(is.numeric, funs(replace(.,is.na(.), mean(., na.rm = TRUE)))) %>%
  mutate_if(is.factor, funs(replace(.,is.na(.), var_mode(na.omit(.)))))


#count remaining missing
#colSums(is.na(data))
```

(a) OLS Model

Hold-out and Train data

```r
#hold data
hold <- data[1:100,]
#train
train <- data[101:nrow(data),]
paste("Number of observations in train data:", nrow(train),"with", ncol(train),"columns" )
```

```
## [1] "Number of observations in train data: 900 with 68 columns"
```

```r
paste("Number of observations in test data:", nrow(hold),"with", ncol(hold),"columns" )
```

```
## [1] "Number of observations in test data: 100 with 68 columns"
```

```r
#full model
model <- lm(SalePrice ~. ,data = train)
# Stepwise regression model
step.model <- stepAIC(model, direction = "both",
                      trace = FALSE)
#summary of the model
summ(step.model)
```

Observations	900
Dependent variable	SalePrice
Type	OLS linear regression

F(89,810)	165.84
R^2	0.95
Adj. R^2	0.94

Stepwise regression model

```r
BIC(step.model)
```

BIC

```
## [1] -1314.297
```

```r
rmse_lm <- sqrt(mean(step.model$residuals^2))
rmse_lm
```

RMSE

```
## [1] 0.08266142
```

```r
VIF(step.model)
```

VIF

```
##                     GVIF Df GVIF^(1/(2*Df))
## MSZoning       23.737712  3        1.695274
## LotArea         2.420379  1        1.555757
## LotConfig       1.447094  3        1.063529
## Neighborhood 1864.794439 17        1.247944
## Condition1      2.218190  5        1.082929
## BldgType       14.115677  4        1.392236
```

```
## OverallQual     3.849691  1        1.962063
## OverallCond     1.923266  1        1.386819
## YearBuilt       8.363010  1        2.891887
## RoofStyle       1.588499  2        1.122656
## Exterior1st    10.330287  7        1.181508
## ExterQual       3.805491  2        1.396699
## ExterCond       1.977169  2        1.185799
## Foundation      9.642107  3        1.458911
## BsmtCond        1.616542  2        1.127578
## BsmtExposure    2.248349  3        1.144574
## BsmtFinSF1      5.643927  1        2.375695
## BsmtFinSF2      1.639739  1        1.280523
## BsmtUnfSF       4.856965  1        2.203852
## HeatingQC       2.052702  2        1.196965
## CentralAir      2.023201  1        1.422393
## Electrical      2.257429  3        1.145343
## X1stFlrSF       5.798951  1        2.408101
## X2ndFlrSF       3.876825  1        1.968965
## LowQualFinSF    1.253629  1        1.119656
## BsmtFullBath    2.073398  1        1.439930
## FullBath        3.149348  1        1.774640
## BedroomAbvGr    2.397242  1        1.548303
## KitchenAbvGr    3.745382  1        1.935299
## Functional      2.544402  5        1.097889
## Fireplaces      1.768861  1        1.329985
## GarageCars      5.837521  1        2.416096
## GarageArea      5.594452  1        2.365259
## GarageCond      1.501465  2        1.106952
## WoodDeckSF      1.383779  1        1.176341
## OpenPorchSF     1.322793  1        1.150128
## EncPorchSF      1.312488  1        1.145639
## PoolArea        1.076190  1        1.037396
## LandSlope       2.805385  2        1.294190
```

```r
lm_rsq <- round(summary(step.model)$r.squared, 5)
paste("R-Squared:", lm_rsq)
```

```
## [1] "R-Squared: 0.94798"
```

```r
#convert categorical to numeric
data[sapply(data, is.factor)] <- data.matrix(data[sapply(data, is.factor)])
#compute correlaiton
correl <- cor(data)
correl[!lower.tri(correl)] <- 0

data.new <-
  data[, !apply(correl, 2, function(x) any(abs(x) > 0.7, na.rm = TRUE))]

#hold data
hold1 <- data.new[1:100,]
#train
train1 <- data.new[101:nrow(data.new),]
paste("Number of observations in train data:", nrow(train1),"with", ncol(train1),"columns" )
```

Stepwise regression model after removing collinear variables

```
## [1] "Number of observations in train data: 900 with 58 columns"
```

```
paste("Number of observations in test data:", nrow(hold1),"with", ncol(hold1),"columns" )
```

```
## [1] "Number of observations in test data: 100 with 58 columns"
```

```
#convert categorical to numeric
data[sapply(data, is.factor)] <- data.matrix(data[sapply(data, is.factor)])
datax <- data
datax$`GarageArea:PoolArea` <- datax$GarageArea*datax$PoolArea
datax$`YearBuilt:YearRemodAdd` <- datax$YearBuilt*datax$YearRemodAdd
datax$`BedroomAbvGr:KitchenAbvGr` <- datax$BedroomAbvGr*datax$KitchenAbvGr
datax$`OverallQual:OverallCond` <- datax$OverallQual*datax$OverallCond
#hold data
hold2 <- datax[1:100,]
#train
train2 <- datax[101:nrow(data),]
paste("Number of observations in train data:", nrow(train2),"with", ncol(train2),"columns" )
```

Stepwise regression model with interaction effects

```
## [1] "Number of observations in train data: 900 with 72 columns"
```

```
paste("Number of observations in test data:", nrow(hold2),"with", ncol(hold2),"columns" )
```

```
## [1] "Number of observations in test data: 100 with 72 columns"
```

```
#full model
model2 <- lm(SalePrice ~. ,data = train2)
# Stepwise regression model
step.model2 <- stepAIC(model2, direction = "both",
                       trace = FALSE)
#summary of the model
summ(step.model2)
```

Report coefficient estimates, p-values, and adjusted R2 for the best model, AIC The stepwise regression model has the highest adjusted R-Squared of 0.93 and with R-Squared of 0.93, the model explains 93% of the variation in the data. However, the model with interaction effects has the lowest AIC as shown below hence the best model of the three.

```
paste("AIC- Stepwise:" ,round(extractAIC(step.model)[2], 4))
```

```
## [1] "AIC- Stepwise: -4307.4041"
```

```
paste("AIC- Interaction Effects:" ,round(extractAIC(step.model2)[2], 4))
```

```
## [1] "AIC- Interaction Effects: -4180.7299"
```

The following table shows the coefficient estimates(`Est.`), p-values (`p`), and adjusted R2 (`Adj. R^2`) for the stepwise regression model with interactions.

```
#summary of the model
summ(step.model2)
```

```
BIC(step.model2)
```

BIC

```
## [1] -1403.73
```

```
rmse_lm2 <- sqrt(mean(step.model2$residuals^2))
rmse_lm2
```

RMSE

```
## [1] 0.09323541
```

```
VIF(step.model2)
```

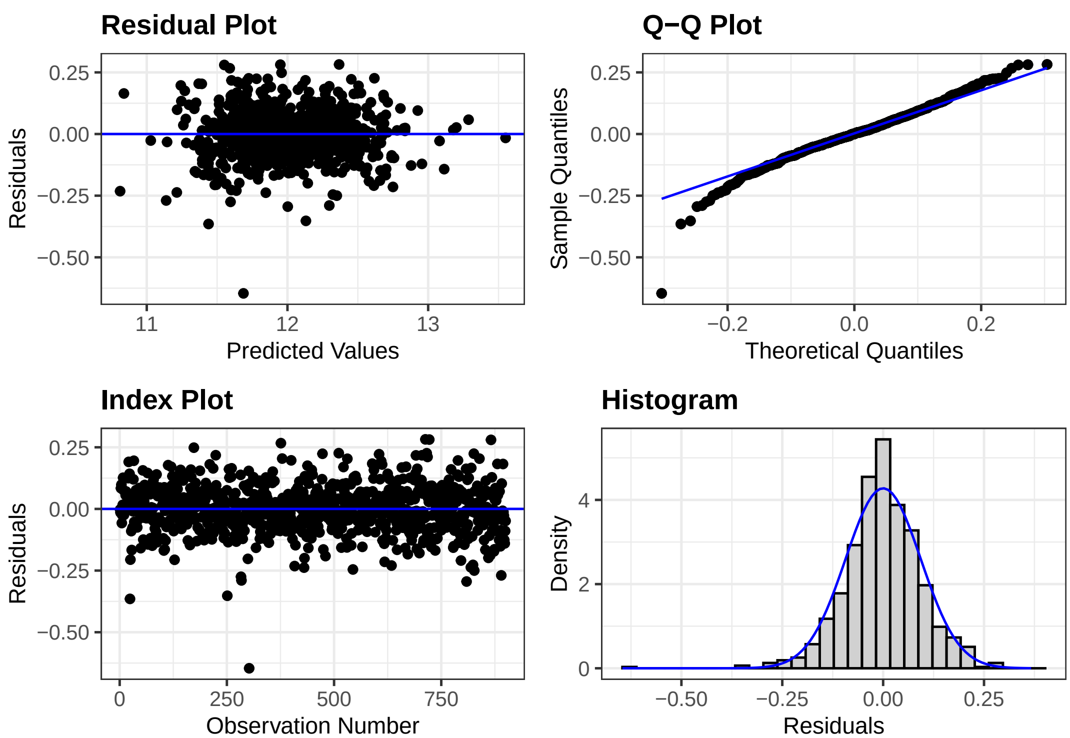
VIF

```
##              MSZoning           LotFrontage               LotArea
##              1.306216              1.595231              1.266480
##              LotShape          Neighborhood            Condition1
##              1.219197              1.229411              1.097026
##              BldgType           OverallQual           OverallCond
##              1.797310             37.309777             27.034811
##          YearRemodAdd             RoofStyle           Exterior1st
##              6.397126              1.171822              3.426345
##           Exterior2nd            MasVnrType             ExterQual
##              3.309035              1.115040              2.772420
##             ExterCond            Foundation              BsmtCond
##              1.232235              2.587254              1.187280
##          BsmtExposure            BsmtFinSF1            BsmtFinSF2
##              1.320666              4.431384              1.379183
##             BsmtUnfSF             HeatingQC            CentralAir
##              3.546428              1.566956              1.562963
##             X1stFlrSF             X2ndFlrSF          LowQualFinSF
##              5.275174              4.236770              1.150247
##          BsmtFullBath          BedroomAbvGr          KitchenAbvGr
##              1.933157              2.521240              1.676046
##           KitchenQual          TotRmsAbvGrd            Functional
##              2.485059              5.293264              1.249885
##            Fireplaces           GarageYrBlt          GarageFinish
```

```
##                1.639889               3.451448               1.804948
##              GarageCars             GarageArea             WoodDeckSF
##                5.311613               5.054878               1.268350
##             OpenPorchSF            EncPorchSF               PoolArea
##                1.201710               1.227207               1.042685
##  `YearBuilt:YearRemodAdd` `OverallQual:OverallCond`
##               13.570155              51.949618
```

```
lm_rsq2 <- round(summary(step.model2)$r.squared, 5)
paste("R-Squared:", lm_rsq2)
```

```
## [1] "R-Squared: 0.93381"
```



Analysis of the residuals

Based on residual plots above, we note that the residuals tend to follow a normal distribution (from the histogram). However, there are some extreme values as shown in the index plot. This might influence our decision to conduct outliear removal before further modeling.

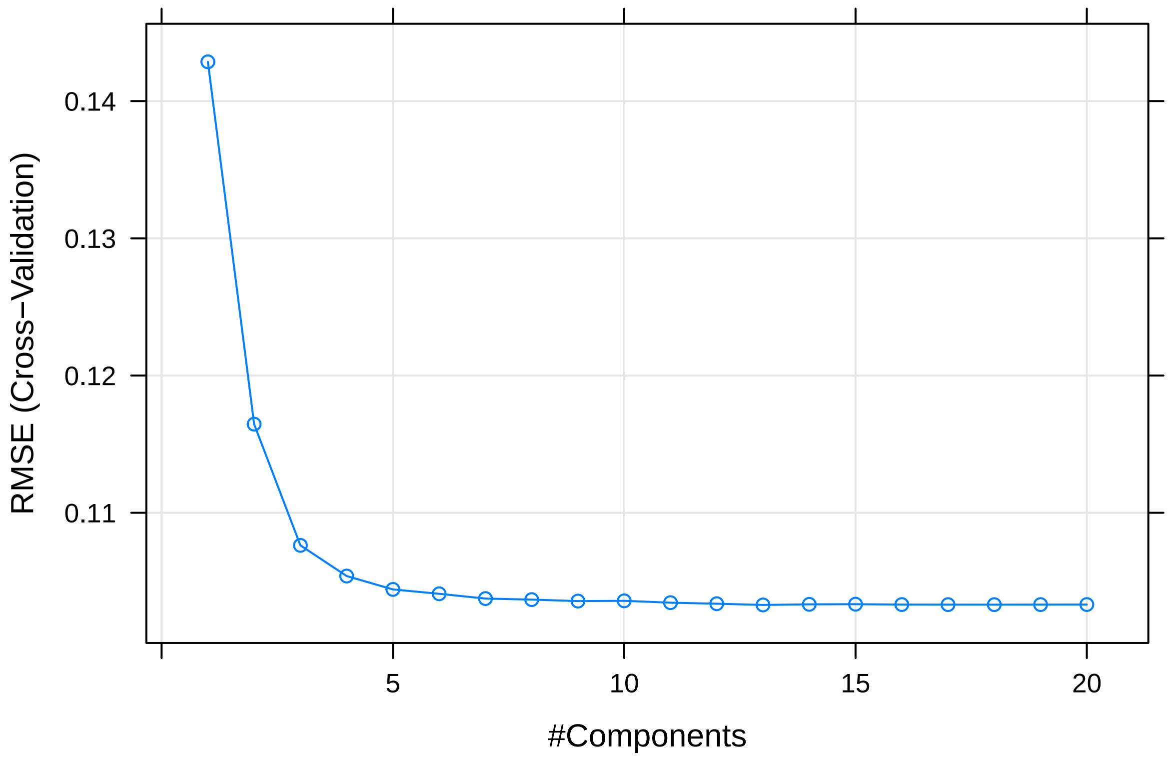
(b) PLS model to predict the log of the sale price.

```
set.seed(1)
plsFit1 <- train(SalePrice~., data=data, method = "pls",
                 tuneLength=20, metric="RMSE",
                 trControl=(trainControl(method="cv", number=5
                 )),
                 preProc=c("center","scale"))
plsFit1
```

```
## Partial Least Squares
##
```

```
## 1000 samples
##   67 predictor
##
## Pre-processing: centered (67), scaled (67)
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 801, 801, 800, 799, 799
## Resampling results across tuning parameters:
##
##   ncomp  RMSE       Rsquared   MAE
##    1     0.1428624  0.8467598  0.10702737
##    2     0.1164585  0.8986846  0.08660085
##    3     0.1076250  0.9136403  0.08125174
##    4     0.1053908  0.9174558  0.08058844
##    5     0.1044146  0.9190081  0.07943962
##    6     0.1040965  0.9194974  0.07880601
##    7     0.1037492  0.9201753  0.07838806
##    8     0.1036735  0.9203394  0.07810993
##    9     0.1035663  0.9205130  0.07809902
##   10     0.1035844  0.9204837  0.07809258
##   11     0.1034502  0.9207373  0.07802095
##   12     0.1033734  0.9208533  0.07795069
##   13     0.1032844  0.9210037  0.07788218
##   14     0.1033284  0.9209129  0.07786214
##   15     0.1033378  0.9208938  0.07787140
##   16     0.1033121  0.9209255  0.07785808
##   17     0.1033078  0.9209335  0.07785178
##   18     0.1033058  0.9209373  0.07784547
##   19     0.1033102  0.9209304  0.07784807
##   20     0.1033133  0.9209249  0.07785008
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was ncomp = 13.
```

```
plot(plsFit1)
```



Chart

```
res <- plsFit1$results
best_perf <- subset(res, res$RMSE == min(res$RMSE) )

pls_perf <- best_perf[,1:3]

pls_perf
```

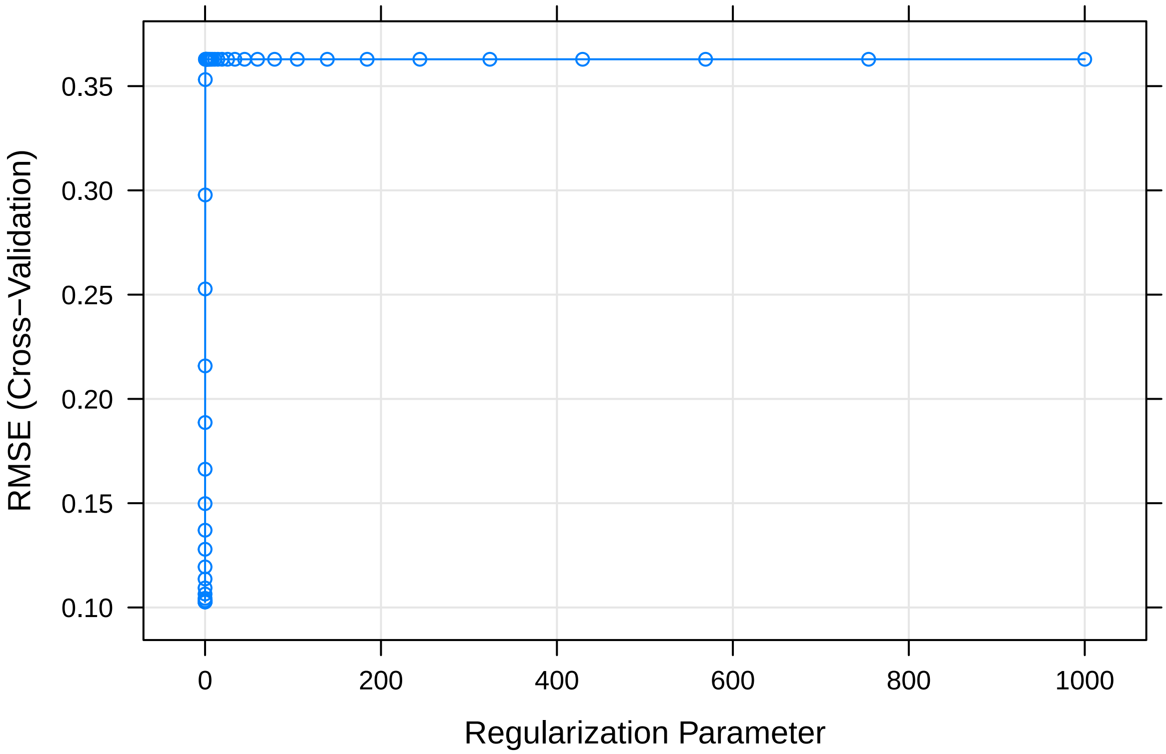
Number of components and the CV RMSE estimate for the final model

```
##    ncomp      RMSE  Rsquared
## 13    13 0.1032844 0.9210037
```

(c) LASSO model to predict the log of the sale price

```
# Build the model
set.seed(1)
lambda <- 10^seq(-3, 3, length = 50)
lasso_caret <- train(
  SalePrice~., data = data, method = "glmnet",
  trControl = trainControl(method = "cv", number = 5),
  tuneGrid = expand.grid(alpha = 1, lambda = lambda)
  )
```

```
plot(lasso_caret)
```



Chart

```
res_lasso <- lasso_caret$results

best_perf_lasso <- subset(res_lasso, res_lasso$RMSE == min(res_lasso$RMSE) )

lasso_perf <- best_perf_lasso[,1:4]
#best
lasso_perf
```

Best fraction and the CV RMSE estimate for the final model

```
##   alpha lambda      RMSE  Rsquared
## 1     1  0.001 0.1026033 0.9220277
```

```
# Model coefficients
paste("Variables with non-zero coefficients with the coefficient values")
```

```
## [1] "Variables with non-zero coefficients with the coefficient values"
```

```
coef(lasso_caret$finalModel, lasso_caret$bestTune$lambda)
```

```
## 68 x 1 sparse Matrix of class "dgCMatrix"
##                           s1
## (Intercept)   9.562309e+00
## MSSubClass   -5.259451e-05
## MSZoning     -4.504714e-02
## LotFrontage   1.927082e-04
## LotArea       2.306189e-06
```

```
## LotShape     -3.628933e-03
## LandContour  -3.644229e-03
## LotConfig     .
## LandSlope     3.216202e-03
## Neighborhood -3.085258e-03
## Condition1    3.277983e-03
## BldgType     -6.511714e-03
## HouseStyle    .
## OverallQual   5.799571e-02
## OverallCond   4.286935e-02
## YearBuilt     1.801703e-03
## YearRemodAdd  5.853094e-04
## RoofStyle     1.715898e-02
## Exterior1st  -4.668259e-03
## Exterior2nd   4.223985e-03
## MasVnrType    7.984307e-03
## MasVnrArea    9.364124e-06
## ExterQual    -1.430859e-02
## ExterCond     1.141894e-02
## Foundation    1.242744e-02
## BsmtQual     -3.178249e-03
## BsmtCond     -2.717721e-02
## BsmtExposure -6.228190e-03
## BsmtFinType1 -1.175198e-03
## BsmtFinSF1    7.212014e-05
## BsmtFinType2  8.954021e-04
## BsmtFinSF2    3.131128e-05
## BsmtUnfSF     .
## TotalBsmtSF   1.131167e-04
## Heating       2.083362e-02
## HeatingQC    -1.371025e-02
## CentralAir    4.525208e-02
## Electrical    .
## X1stFlrSF     3.356214e-06
## X2ndFlrSF     .
## LowQualFinSF -7.509337e-05
## GrLivArea     2.640349e-04
## BsmtFullBath  1.842829e-02
## BsmtHalfBath  5.951212e-03
## FullBath      6.189700e-03
## HalfBath      4.504353e-03
## BedroomAbvGr -1.190067e-02
## KitchenAbvGr -4.979870e-02
## KitchenQual  -2.170358e-02
## TotRmsAbvGrd  5.109794e-03
## Functional    2.427497e-02
## Fireplaces    3.588392e-02
## GarageType    .
## GarageYrBlt  -2.899040e-04
## GarageFinish -9.613169e-03
## GarageCars    3.865621e-02
## GarageArea    8.052745e-05
## GarageQual   -1.580944e-02
## GarageCond   -4.312281e-03
## PavedDrive    1.613583e-02
## WoodDeckSF    5.474086e-05
```

```
## OpenPorchSF   9.826977e-05
## EncPorchSF    1.588360e-04
## PoolArea      1.094606e-04
## MiscVal      -1.320953e-05
## MoSold       -2.437550e-04
## YrSold       -1.472328e-03
## SaleType      .
```

(d) Combination of regression models with missing value imputation

Ridge regression

The data was originally imputed for missing values

```
set.seed(1)

lambda <- 10^seq(-3, 3, length = 5)

# Build the model
set.seed(123)
ridge <- train(
  SalePrice~., data = data, method = "glmnet",
  trControl = trainControl(method = "cv", number = 5),
  tuneGrid = expand.grid(alpha = 0, lambda = lambda)
  )


ridge
```

```
## glmnet
##
## 1000 samples
##   67 predictor
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 799, 801, 800, 802, 798
## Resampling results across tuning parameters:
##
##   lambda        RMSE       Rsquared   MAE
##   1.000000e-03  0.1024033  0.9179538  0.07695364
##   3.162278e-02  0.1024547  0.9179057  0.07699672
##   1.000000e+00  0.1463295  0.8884069  0.10481436
##   3.162278e+01  0.3314160  0.8461505  0.25803359
##   1.000000e+03  0.3615551        NaN  0.28402043
##
## Tuning parameter 'alpha' was held constant at a value of 0
## RMSE was used to select the optimal model using the smallest value.
## The final values used for the model were alpha = 0 and lambda = 0.001.
```

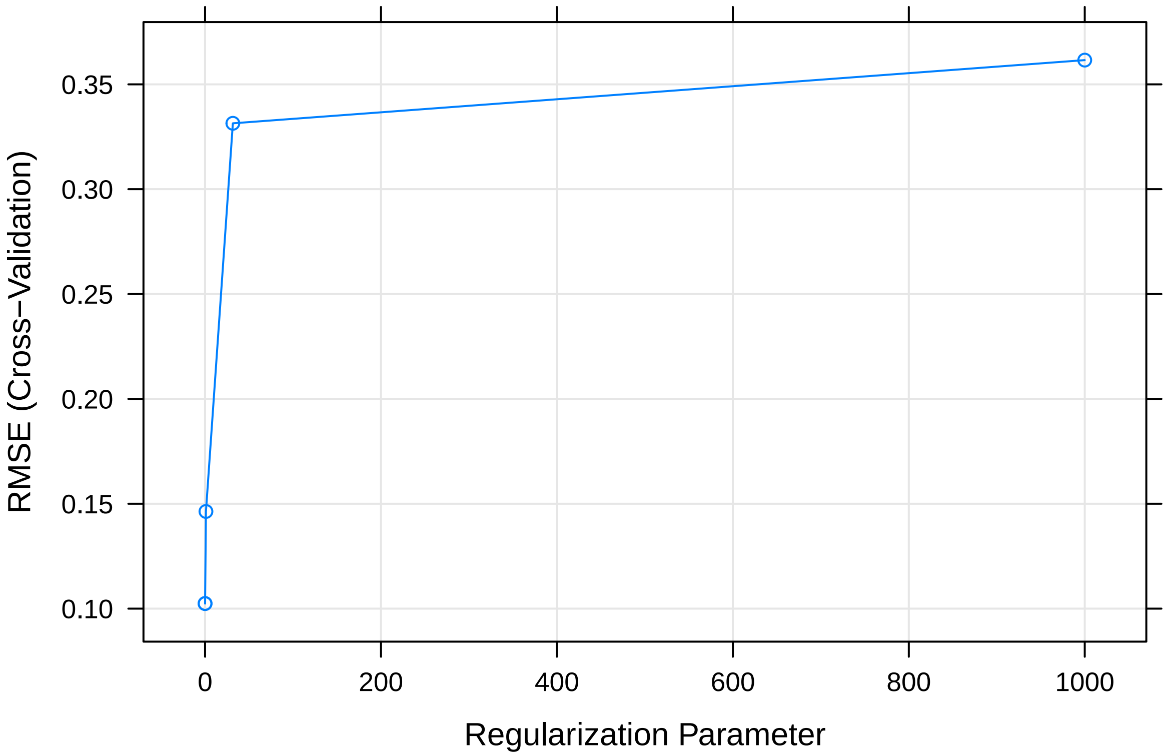
```
res_ridge <- ridge$results

best_perf_ridge <- subset(res_ridge, res_ridge$RMSE == min(res_ridge$RMSE) )

ridge_perf <- best_perf_ridge[,1:4]
ridge_perf
```

```
##    alpha lambda      RMSE  Rsquared
## 1      0  0.001 0.1024033 0.9179538
```

```r
plot(ridge)
```



Chart

```r
# Build the model
set.seed(123)
elastic <- train(
  SalePrice~., data = data, method = "glmnet",
  trControl = trainControl(method = "cv", number = 5),
  tuneLength = 10
  )

#elastic
```

Elastinet

```r
res_elastic <- elastic$results

best_perf_elastic <- subset(res_elastic, res_elastic$RMSE == min(res_elastic$RMSE) )

elastic_perf <- best_perf_elastic[,1:4]

elastic_perf
```

Performance

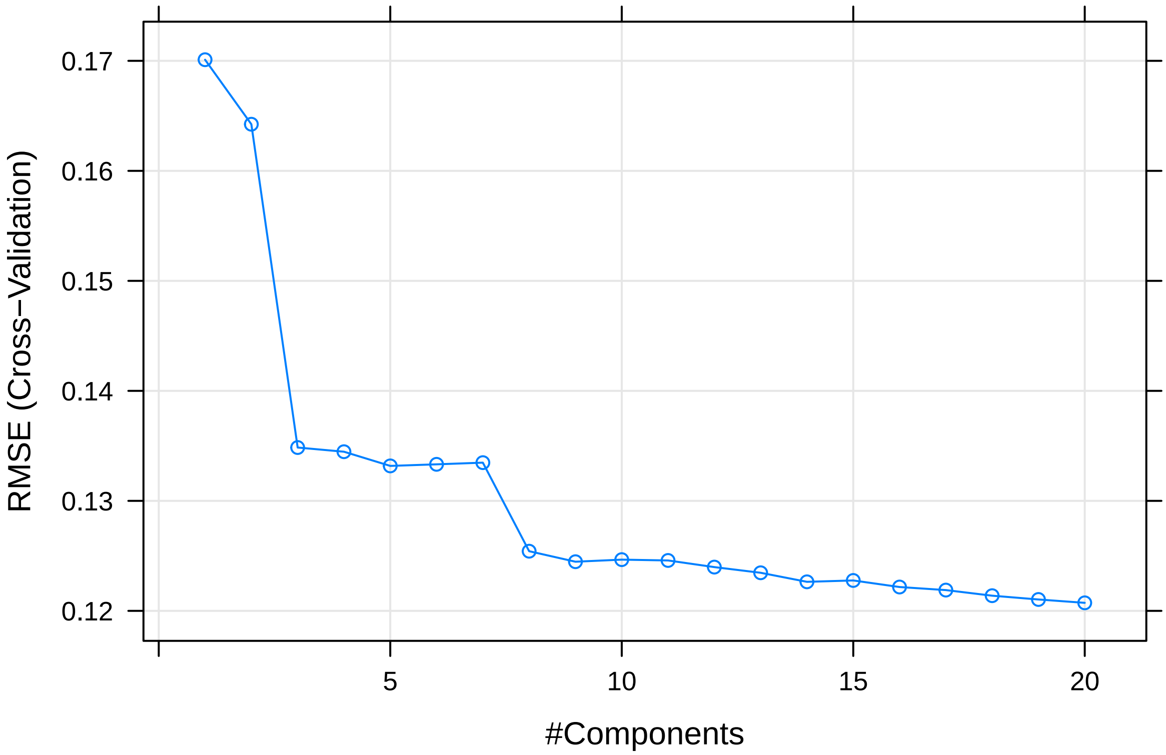
```
##    alpha      lambda     RMSE  Rsquared
## 65   0.7 0.003859544 0.101711 0.9193239
```

```
set.seed(1)
pcrFit <- train(SalePrice~., data=data, method = "pcr",
                tuneLength=20, metric="RMSE",
                trControl=(trainControl(method="cv", number=5
                )),
                preProc=c("center","scale"))
pcrFit
```

PCR

```
## Principal Component Analysis
##
## 1000 samples
##   67 predictor
##
## Pre-processing: centered (67), scaled (67)
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 801, 801, 800, 799, 799
## Resampling results across tuning parameters:
##
##   ncomp  RMSE       Rsquared   MAE
##    1     0.1701037  0.7819618  0.12746462
##    2     0.1642354  0.7972012  0.12384750
##    3     0.1348489  0.8638040  0.10024819
##    4     0.1344690  0.8646946  0.09968538
##    5     0.1331776  0.8672175  0.09883725
##    6     0.1333219  0.8668801  0.09889975
##    7     0.1334738  0.8666219  0.09882253
##    8     0.1254227  0.8823501  0.09349766
##    9     0.1244691  0.8841165  0.09272263
##   10     0.1246582  0.8835870  0.09292842
##   11     0.1245832  0.8838489  0.09277726
##   12     0.1239785  0.8853334  0.09257153
##   13     0.1234621  0.8862226  0.09236980
##   14     0.1226404  0.8878841  0.09166850
##   15     0.1227633  0.8876754  0.09184973
##   16     0.1221681  0.8887609  0.09141195
##   17     0.1218901  0.8893217  0.09131148
##   18     0.1213793  0.8901707  0.09108105
##   19     0.1210364  0.8908090  0.09084028
##   20     0.1207309  0.8914422  0.09070347
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was ncomp = 20.
```

```
plot(pcrFit)
```



Chart

```
res <- pcrFit$results
best_perf <- subset(res, res$RMSE == min(res$RMSE) )

pcr_perf <- best_perf[,1:3]

pcr_perf
```

Model performance

```
##    ncomp      RMSE  Rsquared
## 20    20 0.1207309 0.8914422
```

Table for Model Comparison

```
#Model
Model <- c("OLS", "OLS", "PLS", "LASSO", "Ridge", "elasticNet", "PCR")

#Notes
notes <- c("lm- Stepwise", "lm + 2-way interactions", "caret", "caret and elasticnet","caret and elasticne

#Hyperparameters
hyps <- c("N/A", "N/A", paste("ncomp =",pls_perf$ncomp), paste("alpha=",lasso_perf$alpha, "and lambda =",
```

```r
#RMSE
rmses <- c(rmse_lm, rmse_lm2, pls_perf$RMSE,lasso_perf$RMSE, ridge_perf$RMSE, elastic_perf$RMSE, pcr_perf$
#R-squared
rsqs <- c(lm_rsq, lm_rsq2, pls_perf$Rsquared, lasso_perf$Rsquared, ridge_perf$Rsquared, elastic_perf$Rsqua

#add to dataframe

perf <- data.frame(Model)
perf$Notes <- notes
#hypeparameter
perf$Hyperparameters <- hyps
#RMSE
perf$`CV RMSE` <- rmses
#R-Squared
perf$`CV R2` <- rsqs
#sort by RMSE
perf <- perf[order(perf$`CV RMSE`),]
row.names(perf) <- NULL
perf <- data.frame(lapply(perf, function(y) if(is.numeric(y)) round(y, 4) else y))

knitr::kable(perf, caption = "Table 1: Summary of Model Performance with 5-fold CV")
```

Table 1: Table 1: Summary of Model Performance with 5-fold CV

Model	Notes	Hyperparameters	CV.RMSE	CV.R2
OLS	lm- Stepwise	N/A	0.0827	0.9480
OLS	lm + 2-way interactions	N/A	0.0932	0.9338
elasticNet	caret and elasticnet	alpha = 0.7 and lambda = 0.0039	0.1017	0.9193
Ridge	caret and elasticnet	alpha = 0 and lambda = 0.001	0.1024	0.9180
LASSO	caret and elasticnet	alpha= 1 and lambda = 0.001	0.1026	0.9220
PLS	caret	ncomp = 13	0.1033	0.9210
PCR	caret	ncomp = 20	0.1207	0.8914

	Est.	S.E.	t val.	p
(Intercept)	5.37	0.57	9.38	0.00
MSZoning2	-0.03	0.05	-0.61	0.54
MSZoning3	0.01	0.04	0.30	0.76
MSZoning4	-0.07	0.04	-1.73	0.08
LotArea	0.00	0.00	5.72	0.00
LotConfig2	0.02	0.01	1.70	0.09
LotConfig3	-0.01	0.01	-1.08	0.28
LotConfig4	-0.02	0.02	-0.91	0.36
Neighborhood2	-0.02	0.03	-0.75	0.45
Neighborhood3	-0.07	0.02	-2.80	0.01
Neighborhood4	0.09	0.03	3.46	0.00
Neighborhood5	-0.09	0.02	-4.35	0.00
Neighborhood6	-0.08	0.03	-3.16	0.00
Neighborhood7	0.05	0.03	1.55	0.12
Neighborhood8	-0.10	0.03	-3.86	0.00
Neighborhood9	-0.07	0.02	-3.31	0.00
Neighborhood10	-0.06	0.03	-1.86	0.06
Neighborhood11	0.01	0.03	0.40	0.69
Neighborhood12	-0.07	0.02	-2.95	0.00
Neighborhood13	-0.07	0.02	-3.43	0.00
Neighborhood14	-0.06	0.02	-2.45	0.01
Neighborhood15	-0.08	0.02	-3.39	0.00
Neighborhood16	-0.08	0.03	-3.26	0.00
Neighborhood17	0.01	0.04	0.13	0.90
Neighborhood18	-0.04	0.03	-1.24	0.22
Condition12	0.01	0.02	0.30	0.76
Condition13	0.04	0.02	2.27	0.02
Condition14	-0.02	0.05	-0.42	0.68
Condition15	-0.00	0.03	-0.01	0.99
Condition16	-0.01	0.03	-0.20	0.84
BldgType2	0.03	0.03	0.98	0.33
BldgType3	-0.02	0.03	-0.83	0.41
BldgType4	-0.10	0.02	-4.74	0.00
BldgType5	-0.04	0.02	-2.51	0.01
OverallQual	0.05	0.00	11.40	0.00
OverallCond	0.05	0.00	13.78	0.00
YearBuilt	0.00	0.00	9.36	0.00
RoofStyle2	0.02	0.01	1.75	0.08
RoofStyle3	0.09	0.02	4.06	0.00
Exterior1st2	-0.07	0.02	-2.94	0.00
Exterior1st3	-0.07	0.02	-3.85	0.00
Exterior1st4	-0.06	0.02	-3.30	0.00
Exterior1st5	-0.04	0.02	-2.05	0.04
Exterior1st6	-0.07	0.02	-3.75	0.00
Exterior1st7	-0.05	0.02	-2.73	0.01
Exterior1st8	-0.07	0.02	-3.95	0.00
ExterQual2	-0.01	0.01	-0.97	0.33
ExterQual3	-0.14	0.04	-3.15	0.00
ExterCond2	0.03	0.01	2.55	0.01
ExterCond3	0.05	0.03	1.60	0.11
Foundation2	0.01	0.01	0.94	0.35
Foundation3	-0.02	0.03	-0.76	0.45
Foundation4	0.04	0.02	2.56	0.01
BsmtCond2	-0.02	0.02	-1.17	0.24
BsmtCond3	-0.06	0.03	-2.34	0.02

Observations	900
Dependent variable	SalePrice
Type	OLS linear regression

$F(44,855)$	274.16
R^2	0.93
Adj. R^2	0.93

	Est.	S.E.	t val.	p
(Intercept)	10.72	0.63	16.99	0.00
MSZoning	-0.05	0.01	-7.18	0.00
LotFrontage	0.00	0.00	1.46	0.14
LotArea	0.00	0.00	6.68	0.00
LotShape	-0.00	0.00	-1.92	0.06
Neighborhood	-0.00	0.00	-5.04	0.00
Condition1	0.01	0.00	1.40	0.16
BldgType	-0.01	0.00	-1.94	0.05
OverallQual	0.08	0.01	5.12	0.00
OverallCond	0.07	0.01	4.57	0.00
YearRemodAdd	-0.00	0.00	-4.39	0.00
RoofStyle	0.02	0.01	2.80	0.01
Exterior1st	-0.01	0.00	-2.84	0.00
Exterior2nd	0.01	0.00	2.42	0.02
MasVnrType	0.01	0.01	1.77	0.08
ExterQual	-0.02	0.01	-1.54	0.12
ExterCond	0.02	0.01	2.11	0.03
Foundation	0.01	0.00	2.87	0.00
BsmtCond	-0.04	0.01	-2.69	0.01
BsmtExposure	-0.01	0.00	-1.95	0.05
BsmtFinSF1	0.00	0.00	11.18	0.00
BsmtFinSF2	0.00	0.00	5.92	0.00
BsmtUnfSF	0.00	0.00	7.07	0.00
HeatingQC	-0.01	0.01	-1.89	0.06
CentralAir	0.04	0.02	2.42	0.02
X1stFlrSF	0.00	0.00	13.46	0.00
X2ndFlrSF	0.00	0.00	18.04	0.00
LowQualFinSF	0.00	0.00	2.08	0.04
BsmtFullBath	0.01	0.01	1.61	0.11
BedroomAbvGr	-0.02	0.01	-3.47	0.00
KitchenAbvGr	-0.06	0.02	-3.24	0.00
KitchenQual	-0.02	0.01	-2.32	0.02
TotRmsAbvGrd	0.01	0.00	2.07	0.04
Functional	0.02	0.00	5.47	0.00
Fireplaces	0.03	0.01	5.44	0.00
GarageYrBlt	-0.00	0.00	-1.87	0.06
GarageFinish	-0.01	0.01	-1.47	0.14
GarageCars	0.04	0.01	3.36	0.00
GarageArea	0.00	0.00	2.27	0.02
WoodDeckSF	0.00	0.00	1.91	0.06
OpenPorchSF	0.00	0.00	2.20	0.03
EncPorchSF	0.00	0.00	3.95	0.00
PoolArea	0.00	0.00	1.45	0.15
‘YearBuilt:YearRemodAdd‘	0.00	0.00	8.17	0.00
‘OverallQual:OverallCond‘	-0.00	0.00	-1.49	0.14

Standard errors: OLS

Observations	900
Dependent variable	SalePrice
Type	OLS linear regression

F(44,855)	274.16
R^2	0.93
Adj. R^2	0.93

	Est.	S.E.	t val.	p
(Intercept)	10.72	0.63	16.99	0.00
MSZoning	-0.05	0.01	-7.18	0.00
LotFrontage	0.00	0.00	1.46	0.14
LotArea	0.00	0.00	6.68	0.00
LotShape	-0.00	0.00	-1.92	0.06
Neighborhood	-0.00	0.00	-5.04	0.00
Condition1	0.01	0.00	1.40	0.16
BldgType	-0.01	0.00	-1.94	0.05
OverallQual	0.08	0.01	5.12	0.00
OverallCond	0.07	0.01	4.57	0.00
YearRemodAdd	-0.00	0.00	-4.39	0.00
RoofStyle	0.02	0.01	2.80	0.01
Exterior1st	-0.01	0.00	-2.84	0.00
Exterior2nd	0.01	0.00	2.42	0.02
MasVnrType	0.01	0.01	1.77	0.08
ExterQual	-0.02	0.01	-1.54	0.12
ExterCond	0.02	0.01	2.11	0.03
Foundation	0.01	0.00	2.87	0.00
BsmtCond	-0.04	0.01	-2.69	0.01
BsmtExposure	-0.01	0.00	-1.95	0.05
BsmtFinSF1	0.00	0.00	11.18	0.00
BsmtFinSF2	0.00	0.00	5.92	0.00
BsmtUnfSF	0.00	0.00	7.07	0.00
HeatingQC	-0.01	0.01	-1.89	0.06
CentralAir	0.04	0.02	2.42	0.02
X1stFlrSF	0.00	0.00	13.46	0.00
X2ndFlrSF	0.00	0.00	18.04	0.00
LowQualFinSF	0.00	0.00	2.08	0.04
BsmtFullBath	0.01	0.01	1.61	0.11
BedroomAbvGr	-0.02	0.01	-3.47	0.00
KitchenAbvGr	-0.06	0.02	-3.24	0.00
KitchenQual	-0.02	0.01	-2.32	0.02
TotRmsAbvGrd	0.01	0.00	2.07	0.04
Functional	0.02	0.00	5.47	0.00
Fireplaces	0.03	0.01	5.44	0.00
GarageYrBlt	-0.00	0.00	-1.87	0.06
GarageFinish	-0.01	0.01	-1.47	0.14
GarageCars	0.04	0.01	3.36	0.00
GarageArea	0.00	0.00	2.27	0.02
WoodDeckSF	0.00	0.00	1.91	0.06
OpenPorchSF	0.00	0.00	2.20	0.03
EncPorchSF	0.00	0.00	3.95	0.00
PoolArea	0.00	0.00	1.45	0.15
‘YearBuilt:YearRemodAdd‘	0.00	0.00	8.17	0.00
‘OverallQual:OverallCond‘	-0.00	0.00	-1.49	0.14

Standard errors: OLS