```
%load_ext rpy2.ipython
```

```
from google.colab import files

uploaded = files.upload()

for fn in uploaded.keys():
  print('User uploaded file "{name}" with length {length} bytes'.format(
      name=fn, length=len(uploaded[fn])))
```

选择文件 include_sum_numeric.txt

- **include_sum_numeric.txt**(text/plain) - 4639780 bytes, last modified: 2021/12/11 - 100% done

```
Saving include_sum_numeric.txt to include_sum_numeric.txt
User uploaded file "include_sum_numeric.txt" with length 4639780 bytes
```

```
%%R
include = read.table('include_sum_numeric.txt')
```

```
%%R

install.packages('fastDummies')
library('fastDummies')
```

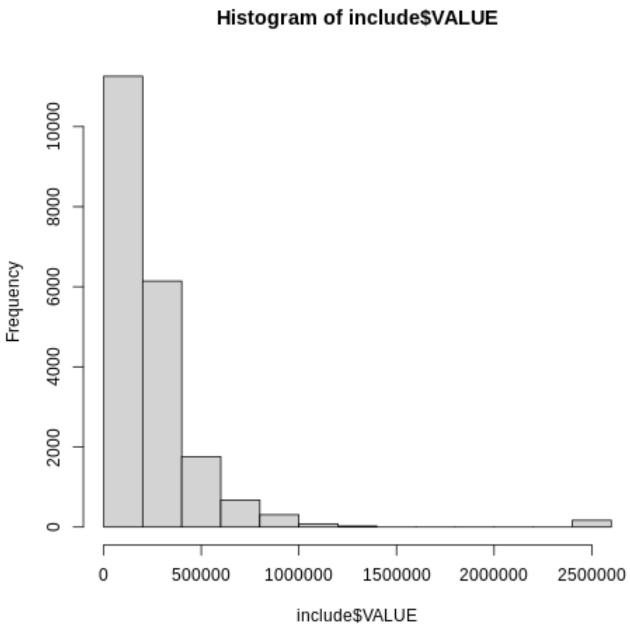
```
%%R
include = dummy_cols(include, select_columns = c('METRO3','SMSA','CMSA','REGION','DIVISION','NUNIT2','BATHS','BEDRMS','BUILT','TENURE','KITCHEN','HHCITSHP','HHRACE','CELLAR','HHGRAD','TYPE
include = subset(include, select = -c(METRO3,SMSA,CMSA,REGION,DIVISION,NUNIT2,BATHS,BEDRMS,BUILT,TENURE,KITCHEN,HHCITSHP,HHRACE,CELLAR,HHGRAD,TYPE,WATER))
```

```
%%R
dim(include)
```

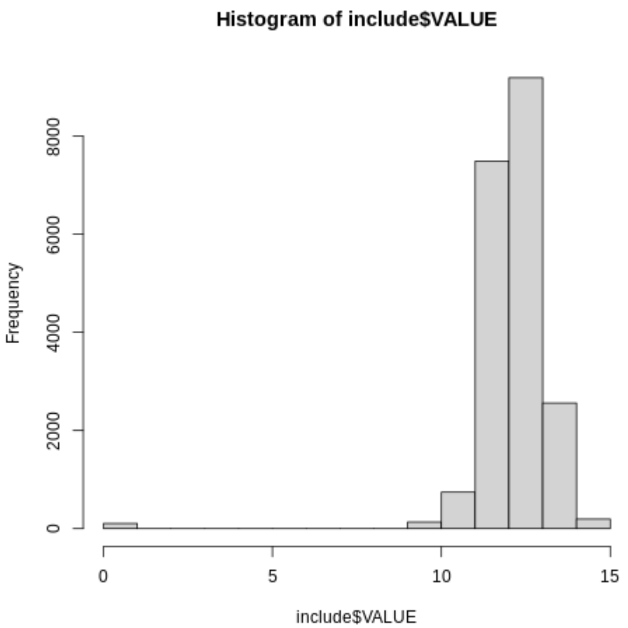
```
[1] 20415   317
```

```
%%R
include = subset(include,select = -CONTROL)
for(i in 1:20415){
    for (j in list('PORCH','IFFEE','TXRE','WATERS','BSINK','SHARPF','TOILET','TUB','GARAGE','DRSHOP','DRSHOP','NOSTEP','OTBUP','EBAR','HDSB','HHSPAN','NEWC','HOTPIP')){
        if (include[i,j]==2){
            include[i,j] = 0
        }
    }
}
for(i in 1:20415){
    if (include[i,'CONDO']==3){
            include[i,'CONDO'] = 0
    }
}
```

```
%%R
hist(include$VALUE)
```




```
%%R
include$VALUE = log(include$VALUE)
hist(include$VALUE)
```




```
%%R
include = subset(include,select = -c(TOILET,NUNITS,CONDO,HOTPIP,TENURE_,TYPE_ ))
```

```
%%R
# 1/5 of sample for prediction in the test
set.seed(0)
n = 20415/5
test_index = sample(c(1:20415),size=n)
sample_test = include[test_index,]
sample_train = include[-test_index,]
```

```
%%R
dim(sample_train)
```

```
[1] 16332   310
```

```
%%R
# linear regression
OLS_model <- glm(VALUE ~., data=sample_train)
summary(OLS_model)
```

```
BUILT_2011     1.409e-01  2.553e-01   0.552 0.580964
BUILT_2012     2.045e-01  2.552e-01   0.801 0.422989
BUILT_2013     4.101e-01  4.912e-01   0.835 0.403774
KITCHEN_2      2.337e-01  1.564e-01   1.494 0.135158
HHCITSHP_2     1.812e-02  9.105e-02   0.199 0.842232
HHCITSHP_3    -2.149e-01  9.194e-02  -2.338 0.019407 *
HHCITSHP_4     2.747e-02  3.602e-02   0.763 0.445755
HHCITSHP_5     2.119e-03  4.558e-02   0.046 0.962915
HHRACE_2      -2.011e-01  3.113e-02  -6.458 1.09e-10 ***
HHRACE_3       2.564e-02  1.184e-01   0.217 0.828568
HHRACE_4      -6.330e-03  4.786e-02  -0.132 0.894773
HHRACE_5      -1.881e-02  1.687e-01  -0.111 0.911222
HHRACE_6      -4.079e-01  1.903e-01  -2.143 0.032124 *
HHRACE_7      -2.552e-01  1.218e-01  -2.095 0.036205 *
HHRACE_8       1.174e-01  2.405e-01   0.488 0.625336
HHRACE_9      -7.933e-01  6.969e-01  -1.138 0.254993
HHRACE_10      2.010e-01  4.920e-01   0.409 0.682848
HHRACE_11     -1.343e-01  5.683e-01  -0.236 0.813137
HHRACE_13      1.478e-01  9.884e-01   0.149 0.881165
HHRACE_14     -4.067e-01  4.233e-01  -0.961 0.336737
HHRACE_15     -5.270e-02  4.025e-01  -0.131 0.895841
HHRACE_17             NA         NA      NA       NA
HHRACE_18      6.118e-01  1.012e+00   0.605 0.545414
HHRACE_19     -6.397e-01  9.876e-01  -0.648 0.517156
HHRACE_21      7.548e-01  9.960e-01   0.758 0.448565
CELLAR_2       4.052e-02  2.513e-02   1.613 0.106848
CELLAR_3      -1.447e-02  2.845e-02  -0.509 0.610988
CELLAR_4      -1.177e-01  3.040e-02  -3.872 0.000108 ***
CELLAR_5      -2.568e-02  8.229e-02  -0.312 0.755013
HHGRAD_32     -2.254e-02  2.828e-01  -0.080 0.936472
HHGRAD_33      2.707e-01  2.624e-01   1.031 0.302382
HHGRAD_34      1.235e-01  2.593e-01   0.476 0.633835
HHGRAD_35      8.527e-02  2.626e-01   0.325 0.745402
HHGRAD_36      1.896e-01  2.603e-01   0.728 0.466534
HHGRAD_37      1.194e-01  2.601e-01   0.459 0.646293
HHGRAD_38      2.820e-01  2.557e-01   1.103 0.270120
HHGRAD_39      2.691e-01  2.506e-01   1.074 0.282918
HHGRAD_40      2.954e-01  2.510e-01   1.177 0.239276
HHGRAD_41      2.437e-01  2.538e-01   0.960 0.336843
HHGRAD_42      2.571e-01  2.527e-01   1.018 0.308930
HHGRAD_43      3.640e-01  2.531e-01   1.438 0.150434
HHGRAD_44      3.399e-01  2.509e-01   1.355 0.175537
HHGRAD_45      4.012e-01  2.515e-01   1.595 0.110719
```

```
HHGRAD_46     2.435e-01  2.558e-01   0.952 0.340980
HHGRAD_47     4.376e-01  2.559e-01   1.710 0.087288 .
WATER_2       5.932e-02  8.868e-02   0.669 0.503566
WATER_3       1.526e-01  2.742e-01   0.556 0.577909
WATER_4       1.130e-01  3.396e-01   0.333 0.739213
WATER_5       9.459e-01  5.682e-01   1.665 0.096014 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for gaussian family taken to be 0.9573114)

    Null deviance: 22577  on 16331  degrees of freedom
Residual deviance: 15349  on 16033  degrees of freedom
AIC: 45934

Number of Fisher Scoring iterations: 2
```

```
%%R
# 9/10 of sample for estimation,  1/10 of sample for test
set.seed(0)
n_est_sample <- round(9*length(sample_train[,1])/10)
est_sample_ind <- sample.int(n = length(sample_train[,1]), size = n_est_sample, replace = FALSE)
training_sample = sample_train[est_sample_ind,]
training_true_y = training_sample['VALUE']
validation_sample = sample_train[-est_sample_ind,]
validation_x = subset(validation_sample, select = -VALUE)
validation_true_y = validation_sample['VALUE']
OLS_model <- glm(VALUE ~ ., data=training_sample)
in_sample_predictions = predict(OLS_model,newdata = training_sample)
out_sample_predictions = predict(OLS_model,newdata = validation_x)
```

```
%%R
is_mse = sum((training_true_y - in_sample_predictions)^2)/length(training_true_y)
is_rmse = is_mse^(0.5)
cat(is_mse,' ', is_rmse)
```

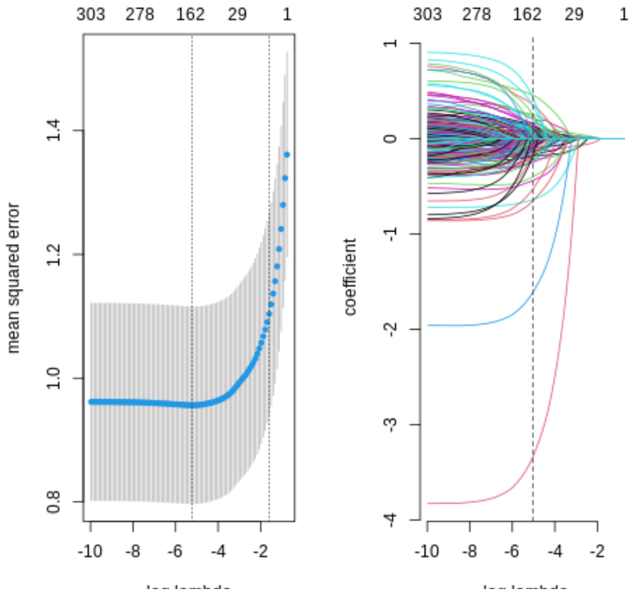
```
13564.91   116.4685
```

```
%%R
oos_mse = sum((validation_true_y - out_sample_predictions)^2)/length(validation_true_y)
oos_rmse = oos_mse^(0.5)
cat(oos_mse,' ', oos_rmse)
```

```
1836.477   42.85414
```

```
%%R
#lasso
install.packages('gamlr')
library(gamlr)
```

```
%%R
set.seed(0)
training_x = subset(training_sample, select = -VALUE)
par(mfrow=c(1,2))
plot(cv_lasso <- cv.gamlr(x=training_x, y=training_sample['VALUE'], lmr=1e-4, nford = 10)
plot(cv_lasso$gamlr)
```




```
%%R
y_predict = drop(predict(cv_lasso, validation_x, select="min"))
head(y_predict)
```

```
       1        5       11       19       28       36
11.86813 12.13239 11.38468 11.93891 12.08198 12.35771
```

```
%%R
nonzero_indice <- c()
for (i in 1:length(colnames(x))){
    if (coef(cv_lasso, select="min")[i] != 0) {
        nonzero_indice <- c(nonzero_indice,i)
    }
}
print(length(nonzero_indice))
colnames(x)[nonzero_indice]
```

```
[1] 142
  [1] "AMTE"          "FAMRM"         "HALFB"         "KITCH"         "OTHFN"
  [6] "LOT"           "UNITSF"        "LPRICE"        "IFFEE"         "HOWN"
 [11] "TXRE"          "WATERS"        "BSINK"         "EXCLUS"        "LAUNDY"
 [16] "OTHRUN"        "DRSHOP"        "FLOORS"        "NOSTEP"        "EBAR"
 [21] "ROOMS"         "ZINC2"         "ZSMHC"         "AMTX"          "VOTHER"
 [26] "HDSB"          "HHSPAN"        "NEWC"          "LMED"          "FMR"
 [31] "WHNGET"        "CSTMNT"        "APPL_SUM"      "HEATCOOL_SUM" "Problem_SUM"
 [36] "DisPLan_SUM"   "METRO3_4"      "SMSA_520"      "SMSA_600"      "SMSA_620"
 [41] "SMSA_720"      "SMSA_1120"     "SMSA_1160"     "SMSA_1680"     "SMSA_1720"
 [46] "SMSA_1920"     "SMSA_2020"     "SMSA_2120"     "SMSA_2160"     "SMSA_2285"
 [51] "SMSA_2400"     "SMSA_2700"     "SMSA_3000"     "SMSA_3280"     "SMSA_3480"
 [56] "SMSA_3560"     "SMSA_4040"     "SMSA_4160"     "SMSA_4280"     "SMSA_4400"
 [61] "SMSA_4880"     "SMSA_4900"     "SMSA_4920"     "SMSA_5015"     "SMSA_5120"
 [66] "SMSA_5160"     "SMSA_5170"     "SMSA_5190"     "SMSA_5360"     "SMSA_5560"
 [71] "SMSA_5640"     "SMSA_5720"     "SMSA_5775"     "SMSA_5880"     "SMSA_5920"
 [76] "SMSA_5960"     "SMSA_6080"     "SMSA_6200"     "SMSA_6840"     "SMSA_6880"
 [81] "SMSA_6920"     "SMSA_7040"     "SMSA_7090"     "SMSA_7160"     "SMSA_7320"
 [86] "SMSA_7360"     "SMSA_7400"     "SMSA_7480"     "SMSA_7500"     "SMSA_7510"
```

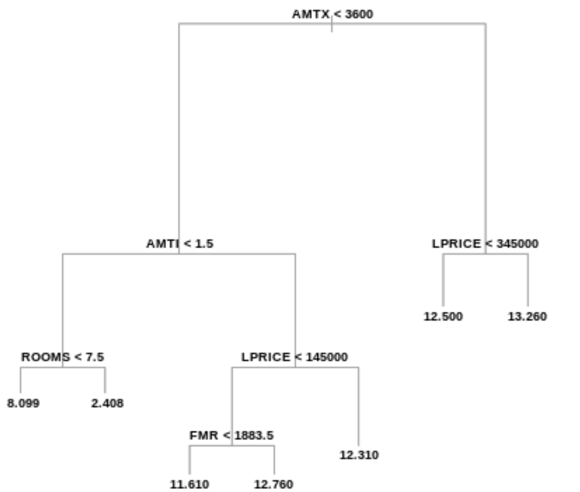
```
 [91] "SMSA_7600"    "SMSA_7840"    "SMSA_8000"    "SMSA_8120"    "SMSA_8280"
 [96] "SMSA_8520"    "SMSA_9240"    "SMSA_9320"    "SMSA_9992"    "SMSA_9993"
[101] "CMSA_10"      "CMSA_34"      "CMSA_41"      "CMSA_47"      "CMSA_49"
[106] "CMSA_78"      "CMSA_82"      "CMSA_91"      "REGION_2"     "DIVISION_3"
[111] "DIVISION_4"   "DIVISION_89"  "BATHS_1"      "BATHS_3"      "BATHS_4"
[116] "BEDRMS_1"     "BEDRMS_3"     "BEDRMS_4"     "BUILT_1920"   "BUILT_1980"
[121] "BUILT_1985"   "BUILT_2000"   "BUILT_2002"   "BUILT_2004"   "BUILT_2008"
[126] "BUILT_2010"   "HHCITSHP_2"   "HHCITSHP_3"   "HHCITSHP_4"   "HHRACE_4"
[131] "HHRACE_9"     "HHRACE_11"    "HHRACE_15"    "HHGRAD_32"    "HHGRAD_36"
[136] "HHGRAD_39"    "HHGRAD_41"    "HHGRAD_42"    "HHGRAD_44"    "HHGRAD_46"
[141] "WATER_3"      "WATER_4"
```

```
%%R
rmse = (sum((validation_sample['VALUE'] - y_predict)^2))^0.5
rmse
```

```
[1] 42.79308
```

```
%%R
#regression tree
install.packages('tree')
library(tree)
```

```
%%R
set.seed(0)

tree <- tree(VALUE ~ ., data=training_sample)

plot(tree, col = 8)
text(tree,cex=.75, font=2)
```



```
%%R
in_sample_predictions = predict(tree,data = training_sample)
out_sample_predictions = predict(tree,data = validation_x)
```

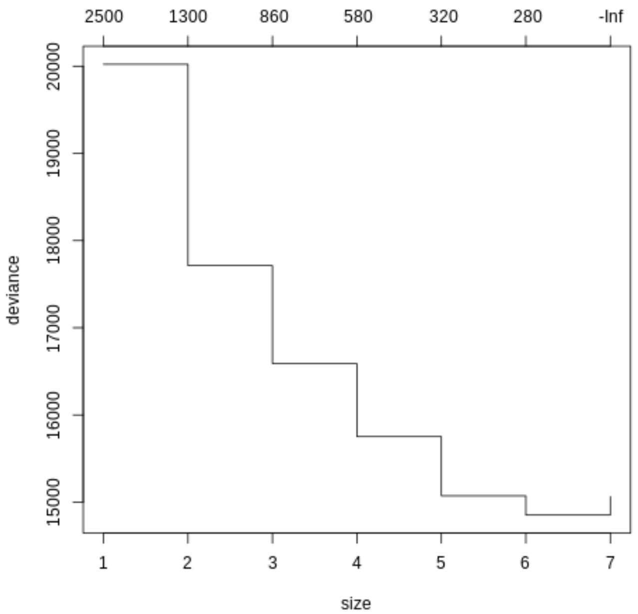
```
%%R
is_mse = sum((training_true_y - in_sample_predictions)^2)/length(training_true_y)
is_rmse = is_mse^(0.5)
cat(is_mse,' ', is_rmse)
```

```
14178.53   119.0736
```

```
%%R
oos_mse = sum((validation_true_y - out_sample_predictions)^2)/length(validation_true_y)
oos_rmse = oos_mse^(0.5)
cat(oos_mse,' ', oos_rmse)
```

```
3095.257   55.63503
```

```
%%R
set.seed(0)
cv_tree <- cv.tree(tree)
plot(cv_tree)
```




```
%%R
cv_tree$dev
```

```
[1] 15062.43 14851.41 15072.18 15751.60 16588.82 17712.77 20022.96
```

```
%%R
```

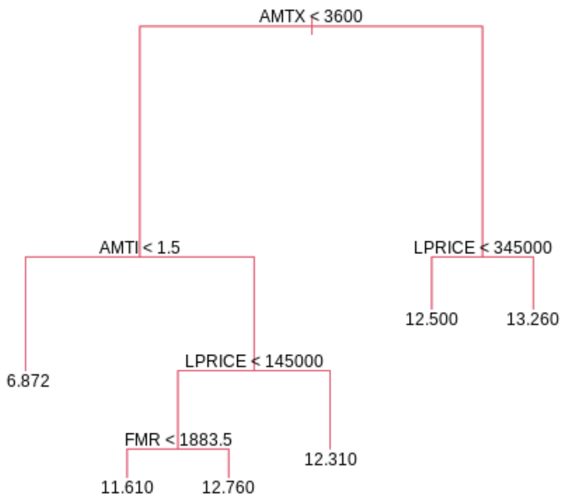
```
cv_tree$size
```

```
    [1] 7 6 5 4 3 2 1
```

```
%%R
( cv_tree_best_size <- cv_tree$size[which.min(cv_tree$dev)] )
```

```
    [1] 6
```

```
%%R
pr_tree <- prune.tree(tree, best=cv_tree_best_size)
plot(pr_tree, col=2)
text(pr_tree)
```



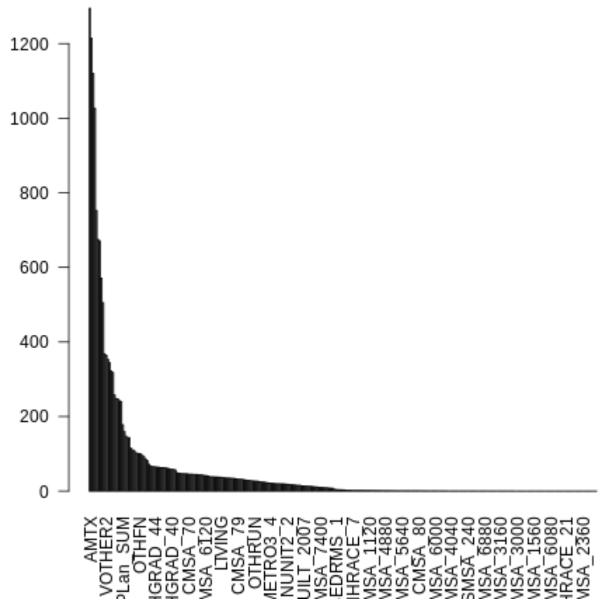

```
%%R
yhat.cvrt <- predict(pr_tree, newdata = validation_x)
oos_rmse = sqrt(sum((validation_sample['VALUE'] - yhat.cvrt)^2))
oos_rmse
```

```
    [1] 44.13945
```

```
%%R
#random forest
set.seed(0)
install.packages('ranger')
library(ranger)
```

```
%%R
rf <- ranger(VALUE ~ ., data=training_sample, write.forest=TRUE, num.tree=200, min.node.size=5, importance="impurity")
barplot(sort(importance(rf), decreasing=TRUE), las=2)
```

```
Growing trees.. Progress: 93%. Estimated remaining time: 2 seconds.
```



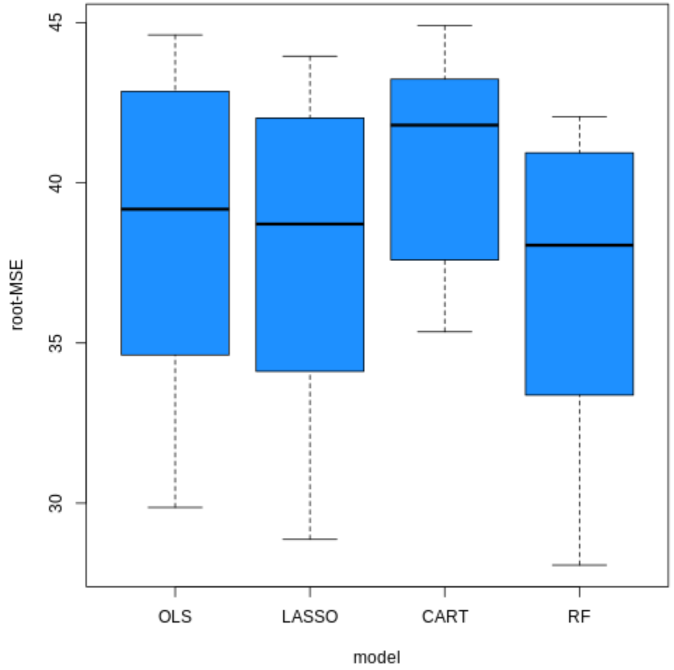

```
%%R
yhat.rf <- predict(rf, data=validation_sample)$predictions
oss_rmse = sqrt(sum((validation_sample['VALUE'] - yhat.rf)^2))
oos_rmse
```

```
[1] 44.13945
```

```
%%R
sample_train_x = subset(sample_train,select = -VALUE)
sample_train_y = sample_train['VALUE']
set.seed(0)
MSE <- list(OLS=NULL, LASSO=NULL, CART=NULL, RF=NULL)
for(i in 1:10){
  train <- sample(1:nrow(sample_train), 14697)

  ols <- glm(VALUE ~ ., data=sample_train[train,])
  yhat.ols <- drop(predict(ols, sample_train_x[-train,]))
  MSE$OLS <- c( MSE$OLS, sqrt(sum((sample_train_y[-train,] - yhat.ols)^2)) )

  lin <- cv.gamlr(x=sample_train_x, y=sample_train_y, lmr=1e-4)
  yhat.lin <- drop(predict(lin, sample_train_x[-train,], select="min"))
  MSE$LASSO <- c( MSE$LASSO, sqrt(sum((sample_train_y[-train,] - yhat.lin)^2)) )

  rt <- tree(VALUE ~ ., data=sample_train[train,])
  #tree <- tree(VALUE ~ ., data=training_sample)
  cvrt <- cv.tree(rt)
  opt_size_rt <- cvrt$size[which.min(cvrt$dev)]
  rtcut <- prune.tree(rt, best=opt_size_rt)
  yhat.cvrt <- predict(rtcut, newdata=sample_train[-train,])
```

```
  MSE$CART <- c( MSE$CART, sqrt(sum((sample_train_y[-train,] - yhat.cvrt)^2)))

  rf <- ranger(VALUE ~ ., data=sample_train[train,],
               num.tree=200, min.node.size=5, write.forest=TRUE)
  yhat.rf <- predict(rf, data=sample_train[-train,])$predictions
  MSE$RF <- c( MSE$RF, sqrt(sum((sample_train_y[-train,] - yhat.rf)^2)) )
}
par(mai=c(.8,.8,.1,.1))
boxplot(as.data.frame(MSE), col="dodgerblue", xlab="model", ylab="root-MSE")
```

```
    Growing trees.. Progress: 99%. Estimated remaining time: 0 seconds.
    Growing trees.. Progress: 100%. Estimated remaining time: 0 seconds.
```




```
%%R
#predict
predict_value = predict(rf, data=sample_test)$predictions
accuracy = 1-abs(exp(sample_test['VALUE']) - exp(predict_value))/exp(sample_test['VALUE'])
summary(accuracy)
```

```
         VALUE
     Min.   :-253614.51
     1st Qu.:      0.64
     Median :      0.79
     Mean   :   -447.79
     3rd Qu.:      0.90
     Max.   :      1.00
```