Lab 8

Housing Data

For this lab, we will consider the housing data again. So far, we have discussed various models for predicting price, but now we will compare various models against each other. As a quick refresher, recall that there are 522 observations with the following variables:

- price: in 2002 dollars
- area: Square footage
- bed: number of bedrooms
- bath: number of bathrooms
- ac: central AC (yes/no)
- garage: number of garage spaces
- pool: yes/no
- year: year of construction
- quality: high/medium/low
- home style: coded 1 through 7
- lot size: sq ft
- highway: near a highway (yes/no)

```
fileName <- "https://raw.githubusercontent.com/ysamwang/btry6020_sp22/main/lectureData/estate.csv"
housing_data <- read.csv(fileName)
head(housing_data)
```

```
##   id  price area bed bath  ac garage pool year quality style   lot highway
## 1  1 360000 3032   4    4 yes      2   no 1972  medium     1 22221      no
## 2  2 340000 2058   4    2 yes      2   no 1976  medium     1 22912      no
## 3  3 250000 1780   4    3 yes      2   no 1980  medium     1 21345      no
## 4  4 205500 1638   4    2 yes      2   no 1963  medium     1 17342      no
## 5  5 275500 2196   4    3 yes      2   no 1968  medium     7 21786      no
## 6  6 248000 1966   4    3 yes      5  yes 1972  medium     1 18902      no
```

```
#create new column for the age of house
housing_data$age <- 2002-housing_data$year
```

Cross Validation

Data Splitting

We will now compare a few different models using cross validation. Notice that each time we include a new variable, the RSS never increases.

```r
# Fix the way of randomly splitting the data,
# such that each time you run will provide same results (optional),
# 1 in set.seed(1) can be set with any number.
set.seed(1)

# Sample splitting 70% training and 30% test
# sample size
n <- dim(housing_data)[1]
m <- floor(n * 0.7)
# generate random training indics (70%)
train_idx <- sample(1:n, m, replace = FALSE)
# extract the training data using training indices
train_set <- housing_data[train_idx,]
# extract the test data
test_set <- housing_data[-train_idx,]
y_true <- log(test_set$price)

#Fit the model using training data
mod1 <- lm(log(price) ~ age + area, data = train_set)
summary(mod1)
```

```
##
## Call:
## lm(formula = log(price) ~ age + area, data = train_set)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.67273 -0.12915  0.00168  0.11301  0.65471
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.168e+01  5.336e-02  218.85   <2e-16 ***
## age         -6.111e-03  6.882e-04   -8.88   <2e-16 ***
## area         4.281e-04  1.608e-05   26.62   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1998 on 362 degrees of freedom
## Multiple R-squared:  0.7807, Adjusted R-squared:  0.7795
## F-statistic: 644.5 on 2 and 362 DF,  p-value: < 2.2e-16
```

```r
#Predict the price of new data (test set) using the model fitted by training set
y_hat_1 <- predict(mod1, test_set)
# Calculate the error: mean squared error (MSE)
pred_error_1 <- mean((y_hat_1-y_true)^2)
pred_error_1
```

```
## [1] 0.05618005
```

```r
# try different model by adding "lot" as explanary variable
mod2 <- lm(log(price) ~ age + area + lot, data = train_set)
summary(mod2)
```

```
##
## Call:
## lm(formula = log(price) ~ age + area + lot, data = train_set)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.62768 -0.10985 -0.00093  0.10410  0.57811
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.163e+01  5.269e-02 220.665  < 2e-16 ***
## age         -6.666e-03  6.757e-04  -9.866  < 2e-16 ***
## area         4.111e-04  1.594e-05  25.790  < 2e-16 ***
## lot          4.505e-06  9.017e-07   4.996 9.13e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1935 on 361 degrees of freedom
## Multiple R-squared:  0.7949, Adjusted R-squared:  0.7932
## F-statistic: 466.4 on 3 and 361 DF,  p-value: < 2.2e-16
```

```r
y_hat_2 <- predict(mod2, test_set)
pred_error_2 <- mean((y_hat_2-y_true)^2)
pred_error_2
```

```
## [1] 0.04740541
```

```r
# try different model by adding "quality" as explanary variable
mod3 <- lm(log(price) ~ age + area + lot + quality, data = train_set)
summary(mod3)
```

```
##
## Call:
## lm(formula = log(price) ~ age + area + lot + quality, data = train_set)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.5989 -0.1085  0.0002  0.1017  0.5008
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    1.202e+01  7.435e-02 161.680  < 2e-16 ***
## age           -4.805e-03  7.155e-04  -6.715 7.38e-11 ***
## area           3.221e-04  1.945e-05  16.558  < 2e-16 ***
## lot            4.496e-06  8.492e-07   5.294 2.09e-07 ***
## qualitylow    -3.664e-01  5.198e-02  -7.049 9.31e-12 ***
## qualitymedium -2.588e-01  3.836e-02  -6.748 6.04e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1814 on 359 degrees of freedom
```

```
## Multiple R-squared:  0.8207, Adjusted R-squared:  0.8182
## F-statistic: 328.6 on 5 and 359 DF,  p-value: < 2.2e-16
```

```r
y_hat_3 <- predict(mod3, test_set)
pred_error_3 <- mean((y_hat_3-y_true)^2)
pred_error_3
```

```
## [1] 0.03438998
```

```r
# try different model by adding "pool" as explanary variable
mod4 <- lm(log(price) ~ age + area + lot + quality + pool, data = train_set)
summary(mod4)
```

```
##
## Call:
## lm(formula = log(price) ~ age + area + lot + quality + pool,
##     data = train_set)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.64483 -0.10591 -0.00114  0.10042  0.50713
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    1.202e+01  7.427e-02 161.900  < 2e-16 ***
## age           -4.858e-03  7.153e-04  -6.791 4.64e-11 ***
## area           3.182e-04  1.960e-05  16.238  < 2e-16 ***
## lot            4.621e-06  8.522e-07   5.423 1.08e-07 ***
## qualitylow    -3.660e-01  5.190e-02  -7.052 9.16e-12 ***
## qualitymedium -2.583e-01  3.830e-02  -6.744 6.22e-11 ***
## poolyes        5.955e-02  4.060e-02   1.467    0.143
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1811 on 358 degrees of freedom
## Multiple R-squared:  0.8218, Adjusted R-squared:  0.8188
## F-statistic: 275.1 on 6 and 358 DF,  p-value: < 2.2e-16
```

```r
y_hat_4 <- predict(mod4, test_set)
pred_error_4 <- mean((y_hat_4 - y_true)^2)
pred_error_4
```

```
## [1] 0.03370149
```

Questions:

1. Compare the models above. Which one is best for prediction? Are there other considerations you would consider when choosing a model?
2. Try on your own: choose different number you like in "set.seed()" function, and re-run the procedure above. Which one is better based on your own results?

K-fold Cross Validation

One-time sample-splitting results above highly depend on the random selection of the training and test set. K-fold Cross Validation is a procedure to alleviate such randomness. Besides, computationally K-fold Cross Validation is much more feasible than leave-one-out Cross Validation (LOOCV).

In the K-fold Cross Validation, here we set K=5. which means we split the data into 5 qual sized subsets. Then for each k=1,...,5, hold out the kth subset and train the model based on the other 4 subsets, calculate kth fold mean square error (MSE) as MSE_k. Finally optain the total MSE by averaging $MSE_k, k = 1, ..., 5$.

K-fold Cross Validation can be done manually, but can also be implemented using "glm" functions.

```
library("boot")
#fit model 1 with age and area
mod_cv_1 <- glm(log(price) ~ age + area, data = housing_data)
summary(mod_cv_1)
```

```
##
## Call:
## glm(formula = log(price) ~ age + area, data = housing_data)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -0.71445  -0.13108  -0.01131   0.10019   1.10462
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.167e+01  4.666e-02  250.09   <2e-16 ***
## age         -6.471e-03  5.850e-04  -11.06   <2e-16 ***
## area         4.389e-04  1.451e-05   30.25   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.04466756)
##
##     Null deviance: 97.083  on 521  degrees of freedom
## Residual deviance: 23.182  on 519  degrees of freedom
## AIC: -136.28
##
## Number of Fisher Scoring iterations: 2
```

```
err_cv_1 <- cv.glm(housing_data, mod_cv_1, K=5)$delta[1]
err_cv_1
```

```
## [1] 0.0454693
```

```
#fit model 2 with age, area and lot
mod_cv_2 <- glm(log(price) ~ age + area + lot, data = housing_data)
summary(mod_cv_2)
```

```
##
## Call:
## glm(formula = log(price) ~ age + area + lot, data = housing_data)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -0.69801  -0.11428  -0.00404   0.10200   0.76684
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.161e+01  4.483e-02 258.927  < 2e-16 ***
## age         -7.323e-03  5.638e-04 -12.988  < 2e-16 ***
## area         4.138e-04  1.409e-05  29.367  < 2e-16 ***
```

```
## lot          6.093e-06  7.734e-07   7.878 1.96e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.03996489)
##
##     Null deviance: 97.083  on 521  degrees of freedom
## Residual deviance: 20.702  on 518  degrees of freedom
## AIC: -193.36
##
## Number of Fisher Scoring iterations: 2
```

```r
err_cv_2 <- cv.glm(housing_data, mod_cv_2, K=5)$delta[1]
err_cv_2
```

```
## [1] 0.04050282
```

Questions:

- Use "names(housing_data)'' to see all the possible covariates you could use.
- Implement 5-fold cross-validation again to choose the covariates in housing_data that you believe might be associated with the house price. Fit new model and compare the error with the results above. Can you find a better one? Given the set of covariates, can you try to search through all possible models to find the best one?

Penalized Scores

Cross validation can be computationally expensive, since it requires refitting the model on many different "test sets.'' On Wednesday, we will discuss potential alternatives to cross validation which don't require sample splitting and only fit the model once. In particular, they assign a score to each model, but explicitly include a penalty for more complex models. In particular, the two scores we will discuss are AIC (Akaike information criterion) and BIC (Bayesian information criterion).

1. R^2 measures how well the fitted model predicts the data it was fitted on, and will always increase when we include additional covariates.
2. Adjusted R^2 add adjustment to penalize for increasing model complexity, but it is still not good enough.
3. AIC, BIC require model assumptions to be theoretically grounded, but work well empirically even when the assumptions don't hold.

Penalized Scores are calculated based on the whole dataset. Next we will use 3 models to compare the choice of the "best" one among three using different model selection criteria. Similar to golf, when R calculates AIC and BIC, a smaller score indicates a better model.

```r
#Create variables to store the criterion results from different models
r_squared <- rep(0,3)
adj_r_squared <- rep(0,3)
aic <- rep(0,3)
bic <- rep(0,3)
cv_error <- rep(0,3)

#fit model 1 with age and area
model1 <- lm(log(price) ~ age + area, data = housing_data)
summary(model1)
```

```
##
```

```
## Call:
## lm(formula = log(price) ~ age + area, data = housing_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.71445 -0.13108 -0.01131  0.10019  1.10462
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.167e+01  4.666e-02  250.09   <2e-16 ***
## age         -6.471e-03  5.850e-04  -11.06   <2e-16 ***
## area         4.389e-04  1.451e-05   30.25   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2113 on 519 degrees of freedom
## Multiple R-squared:  0.7612, Adjusted R-squared:  0.7603
## F-statistic: 827.2 on 2 and 519 DF,  p-value: < 2.2e-16
```

```
sum1 <- summary(model1)
r_squared[1] <- sum1$r.squared
adj_r_squared[1] <- sum1$adj.r.squared
aic[1] <- AIC(model1)
bic[1] <- BIC(model1)

model1_cv <- glm(log(price) ~ age + area, data = housing_data)
cv_error[1] <- cv.glm(housing_data, model1_cv, K=5)$delta[1]

#fit model 2 with age, area and lot
model2 <- lm(log(price) ~ age + area + lot, data = housing_data)
summary(model2)
```

```
##
## Call:
## lm(formula = log(price) ~ age + area + lot, data = housing_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.69801 -0.11428 -0.00404  0.10200  0.76684
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.161e+01  4.483e-02 258.927  < 2e-16 ***
## age         -7.323e-03  5.638e-04 -12.988  < 2e-16 ***
## area         4.138e-04  1.409e-05  29.367  < 2e-16 ***
## lot          6.093e-06  7.734e-07   7.878 1.96e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1999 on 518 degrees of freedom
## Multiple R-squared:  0.7868, Adjusted R-squared:  0.7855
## F-statistic: 637.1 on 3 and 518 DF,  p-value: < 2.2e-16
```

```
sum2 <- summary(model2)
r_squared[2] <- sum2$r.squared
```

```r
adj_r_squared[2] <- sum2$adj.r.squared
aic[2] <- AIC(model2)
bic[2] <- BIC(model2)

model2_cv <- glm(log(price) ~ age + area + lot, data = housing_data)
cv_error[2] <- cv.glm(housing_data, model2_cv, K=5)$delta[1]

#fit model 3 with age, area, lot and ac
model3 <- lm(log(price) ~ age + area + lot + ac, data = housing_data)
summary(model3)
```

```
##
## Call:
## lm(formula = log(price) ~ age + area + lot + ac, data = housing_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.6830 -0.1141 -0.0083  0.1046  0.7096
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.152e+01  5.169e-02 222.823  < 2e-16 ***
## age         -6.654e-03  5.927e-04 -11.226  < 2e-16 ***
## area         4.082e-04  1.405e-05  29.055  < 2e-16 ***
## lot          6.338e-06  7.693e-07   8.238 1.44e-15 ***
## acyes        8.664e-02  2.582e-02   3.356 0.000849 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.198 on 517 degrees of freedom
## Multiple R-squared:  0.7913, Adjusted R-squared:  0.7897
## F-statistic: 490.1 on 4 and 517 DF,  p-value: < 2.2e-16
```

```r
sum3 <- summary(model3)
r_squared[3] <- sum3$r.squared
adj_r_squared[3] <- sum3$adj.r.squared
aic[3] <- AIC(model3)
bic[3] <- BIC(model3)

model3_cv <- glm(log(price) ~ age + area + lot + ac, data = housing_data)
cv_error[3] <- cv.glm(housing_data, model3_cv, K=5)$delta[1]

#compare three models
name = c('model1', 'model2', 'model3')
rbind(name, r_squared, adj_r_squared, aic, bic, cv_error)
```

```
##               [,1]                 [,2]                 [,3]
## name          "model1"             "model2"             "model3"
## r_squared     "0.761209684515937"  "0.786761547422801"  "0.791307691618793"
## adj_r_squared "0.760289490621972"  "0.785526575689728"  "0.789693050934993"
## aic           "-136.277934352758"  "-193.355175026475"  "-202.604315069417"
## bic           "-119.247264001227"  "-172.066837087062"  "-177.058309542121"
## cv_error      "0.0448887517945844" "0.0415066441952207" "0.0403665256509297"
```

Questions

1. Compare the different criterions above, which model do you think is the best?
2. Suppose you have chosen the best one among the three, is that possible we could find a better one other than these three models?