Homework 1

Yiming Gao (NetID: yimingg2)

2017/1/25

Exercise 1

```r
library(tibble)
library(readr)
mydata = read_csv("hw01-data.csv")

# Split the data into training and test set
## 50% of the sample size
smp_size = 0.5 * nrow(mydata)

## Set the seed to make our partition reproductible
set.seed(42)
train_ind = sample(seq_len(nrow(mydata)), size = smp_size)

train = mydata[train_ind, ]
test = mydata[-train_ind, ]
```

Then we fit four models and calculate RMSE and number of parameters for each model. I set the number of digits as 3, for readability.

- Model 1: `y ~ .`
- Model 2: `y ~ . + I(a ^ 2) + I(b ^ 2) + I(c ^ 2)`
- Model 3: `y ~ . ^ 2 + I(a ^ 2) + I(b ^ 2) + I(c ^ 2)`
- Model 4: `y ~ a * b * c * d * I(a ^ 2) * I(b ^ 2) * I(c ^ 2)`

```r
library(Metrics)
model1 = lm(y ~ ., data = train)
model2 = lm(y ~ . + I(a ^ 2) + I(b ^ 2) + I(c ^ 2), data = train)
model3 = lm(y ~ . ^ 2 + I(a ^ 2) + I(b ^ 2) + I(c ^ 2), data = train)
model4 = lm(y ~ a * b * c * d * I(a ^ 2) * I(b ^ 2) * I(c ^ 2), data = train)
```

```r
# Calculate the RMSE and number of parameters
model1.train = round(rmse(train$y, model1$fitted.values),digits = 3)
model1.test = round(rmse(test$y, predict(model1, newdata = test)),digits = 3)
```

```
model2.train = round(rmse(train$y, model2$fitted.values),digits = 3)
model2.test = round(rmse(test$y, predict(model2, newdata = test)),digits = 3)
model3.train = round(rmse(train$y, model3$fitted.values),digits = 3)
model3.test = round(rmse(test$y, predict(model3, newdata = test)),digits = 3)
model4.train = round(rmse(train$y, model4$fitted.values),digits = 3)
model4.test = round(rmse(test$y, predict(model4, newdata = test)),digits = 3)
model1.par = length(model1$coefficients) - 1
model2.par = length(model2$coefficients) - 1
model3.par = length(model3$coefficients) - 1
model4.par = length(model4$coefficients) - 1
```

(a) For each of the models above, report:

- Train RMSE
- Test RMSE
- Number of Parameters, Excluding the Variance

Model	Train RMSE	Test RMSE	Number of Parameters
Model 1	2.995	2.949	4
Model 2	2.084	2.192	7
Model 3	1.021	1.041	13
Model 4	0.898	1.272	127

(b) Based on these results, we may consider Model 4 is overfitting because Test RMSE is much larger than Train RMSE. Probably Model 1 is underfitting, since Train RMSE is unusually greater than Test RMSE, and only four parameters may not be able to explain the relationship between the predictors and response. Model 3 performs the best among 4 models.

(c) I ran the **stepAIC** function in MASS package, to select parameters. The output is redundant, so I only include the final model. The parameters I chose were `a, b, d,a^2, a:b, b:d`.

```
# Find out the predictors
names(step$coefficients)
```

```
## [1] "(Intercept)" "a"           "b"           "d"           "I(a^2)"
## [6] "a:b"         "b:d"
```

```
model.best = lm(y ~ a + b + d + I(a^2) + a:b + b:d, data = train)


# Report Train and Test RMSE
```

```r
best.train.rmse = round(rmse(train$y, model.best$fitted.values),digits = 3)
best.test.rmse = round(rmse(test$y, predict(model.best, newdata = test)),digits = 3)
best.par = length(model.best$coefficients) - 1

# Arrange the information
a = "Best Model"
b = best.train.rmse
c = best.test.rmse
d = best.par
my_data <- data.frame(a, b, c, d)
names(my_data) <- c("Model", "Train RMSE", "Test RMSE","Number of Parameters")
kable(my_data, align = "c", padding=4)
```

Model	Train RMSE	Test RMSE	Number of Parameters
Best Model	1.026	1.041	6

From the table above, we know that Train RMSE and Test RMSE are 1.026 and 1.041, which means the model in neither underfitting or overfitting. Both of them are close to the results of Model 3, but this model has fewer parameters than Model 3. For simplicity and model performance, we should consider this model:

```
y ~ a + b + d + I(a ^ 2) + a:b + b:d
```

Exercise 2

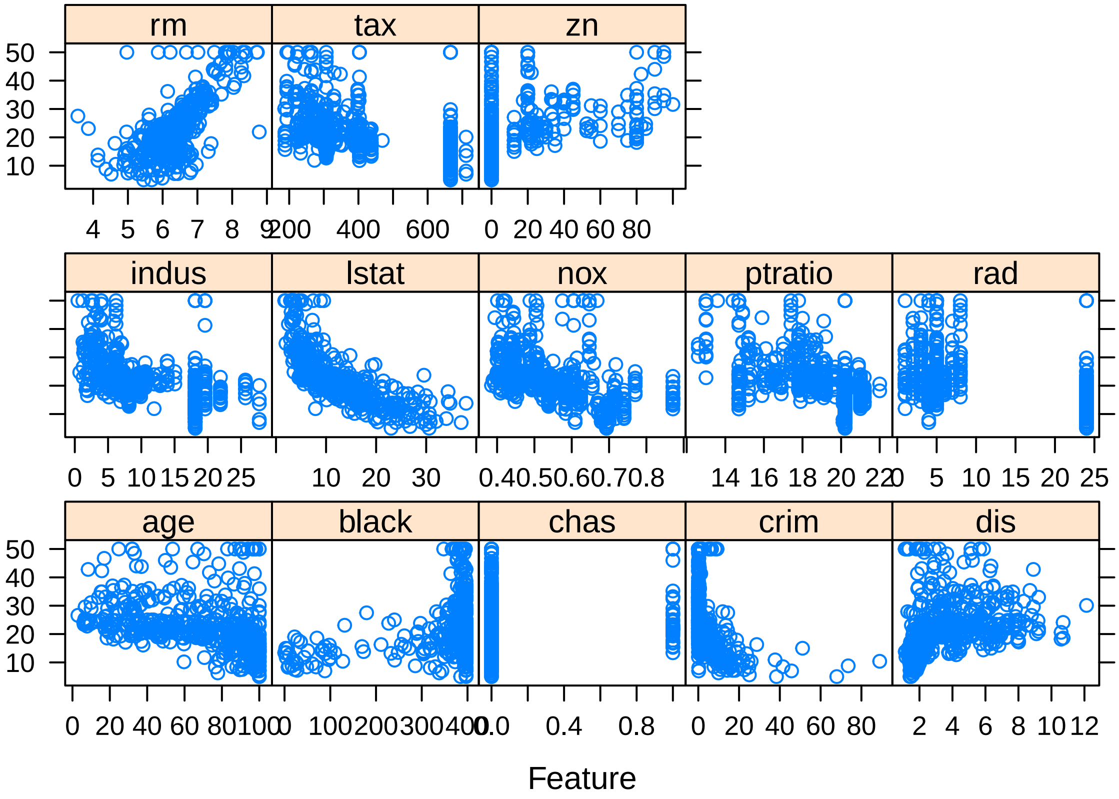
We will use the `Boston` data from the `MASS` package.

First we set seed 314 to control randomization, then randomly split the data into train and test sets using 456 observations for the training data and the remainder for the testing data.

```r
set.seed(314)
train_size = 456
train_index = sample(1:nrow(Boston), size = train_size)
train_data = Boston[train_index, ]
test_data = Boston[-train_index, ]
```

First we draw scatter plots between all predictors and the reponse, in order to explore if there is any significant relationships between them. We will start with `model1` simply with predictors which has absolute correlations with `medv` greater than 0.5, i.e. `ptratio`, `lstat`and `rm`.

```
library(caret)
featurePlot(x = Boston[, c("zn", "indus", "chas", "nox", "rm", "age", "dis", "rad",
                           "tax", "ptratio", "black", "lstat", "crim")],
            plot = "scatter", y = Boston$medv,layout = c(5, 3))
```



```
# We first keep variables with absolute correlations > 0.5
names(Boston)[abs(cor(Boston)[,"medv"])>0.5]
```

```
## [1] "rm"      "ptratio" "lstat"   "medv"
```

We fit out first model with the predictors above, then include all the predictors as our second model.

Get the RMSE of this models for both training and test sets. And arrange the model information into a table. We finally got five nested linear models with `medv` as the response. R code blocks are omitted in the report.

- Model 1: `medv ~ rm + ptratio + lstat`
- Model 2: `medv ~.`
- Model 3: `medv ~ . + I(zn^2) + I(indus^2) + I(chas^2) + I(nox^2) + I(rm^2) + I(age^2) + I(dis^2) + I(rad^2) + I(tax^2) + I(ptratio^2) + I(black^2) + I(lstat^2) + I(crim^2)`

- Model 4: `medv ~ zn*indus*chas*nox*rm*age*dis*rad*tax*ptratio*black*lstat*medv + I(zn^2) + I(indus^2) + I(chas^2) + I(nox^2) + I(rm^2) + I(age^2) + I(dis^2) + I(rad^2) + I(tax^2) + I(ptratio^2) + I(black^2) + I(lstat^2) + I(crim^2)`
- Model 5: `medv ~ I(zn^2)*I(indus^2)*I(chas^2)*I(nox^2)*I(rm^2)*I(age^2)*I(dis^2) *I(rad^2)*I(tax^2)*I(ptratio^2)*I(black^2)*I(lstat^2)*I(crim^2) +"zn*indus*chas*nox*rm*age`

The codes for train RMSE, test RMSE and number of parameters are similar as those in Exercise 1, so I simply omit them, just leave the final results.

Model	Train RMSE	Test RMSE	Number of Parameters
Model 1	5.102	6.109	3
Model 2	4.621	5.270	13
Model 3	3.733	4.388	26
Model 4	0.000	23.513	8203
Model 5	0.000	186.862	16381

From the table above, we can say `Model 1` and `Model 2` are probably underfitting, `Model 4` and `Model 5` are obviously overfitting, because they have low training RMSE and very high test RMSE. So we suppose our best model is

`Model 3: medv ~ . + I(zn^2) + I(indus^2) + I(chas^2) + I(nox^2) + I(rm^2) + I(age^2) + I(dis^2) + I(rad^2) + I(tax^2) + I(ptratio^2) + I(black^2) + I(lstat^2) + I(crim^2)`

Exercise 3

We continue using the Boston data, training split, and models from Exercise 2. The best model we choose is `Model 3`. Refit this model with each of the following modifications:

- `Fit 1` fits the model removing observations with absolute studentized residuals greater than 2.

```
index1 = which(abs(rstudent(model3)) > 2)
train_data1 = train_data[-index1,]
fit1 = lm(medv ~ . + I(zn^2) + I(indus^2) + I(chas^2) + I(nox^2) +
            I(rm^2) + I(age^2) + I(dis^2) + I(rad^2) + I(tax^2) +
            I(ptratio^2) + I(black^2) + I(lstat^2) + I(crim^2),
          data = train_data1)

fit1.test = rmse(test_data$medv, predict(fit1, newdata = test_data))
```

- - `Fit 2` fits the model removing observations with absolute studentized residuals greater than 3.

```r
index2 = which(abs(rstudent(model3)) > 3)
train_data2 = train_data[-index2,]
fit2 = lm(medv ~ . + I(zn^2) + I(indus^2) + I(chas^2) + I(nox^2) +
            I(rm^2) + I(age^2) + I(dis^2) + I(rad^2) + I(tax^2) +
            I(ptratio^2) + I(black^2) + I(lstat^2) + I(crim^2),
          data = train_data2)

fit2.test = rmse(test_data$medv, predict(fit2, newdata = test_data))
```

- `Fit 3` fits the model removing observations with a Cook's distance greater than $\frac{4}{n}$.

```r
index3 = which(cooks.distance(model3) > 4/nrow(train_data))
train_data3 = train_data[-index3,]
fit3 = lm(medv ~ . + I(zn^2) + I(indus^2) + I(chas^2) + I(nox^2) +
            I(rm^2) + I(age^2) + I(dis^2) + I(rad^2) + I(tax^2) +
            I(ptratio^2) + I(black^2) + I(lstat^2) + I(crim^2),
          data = train_data3)

fit3.test = rmse(test_data$medv, predict(fit3, newdata = test_data))
```

(a)

We summarize four models as well as their test RMSE and number of removed observations in the table below.

Model	Test RMSE	Number of Removed Observations
Original model	4.388	0
Fit 1	5.075	12
Fit 2	5.034	5
Fit 3	6.564	24

We can notice that the original model fits the data best. So we cannot simply remove unusual observations from the training data. The results can be even worse. Meanwhile, the more observations we remove, the model has higher test RMSE.

(b) We create a 99% **prediction interval** for an observation with the original model. I set the digits to 3 for readability.

crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	black	lstat
0.02763	75.0	2.95	0	0.4280	6.595	21.8	5.4011	3	252	18.3	395.63	4.32

```r
new_obs = data.frame(crim = 0.02763, zn = 75.0, indus = 2.95, chas = 0, nox = 0.4280,
                     rm = 6.595, age = 21.8, dis = 5.4011, rad = 3, tax = 252,
                     ptratio = 18.3, black = 395.63, lstat = 4.32)
round(predict(model3, newdata = new_obs, interval = "prediction",
              level = 0.99),digits = 3)
```

```
##      fit    lwr    upr
## 1 29.684 19.504 39.863
```

99% prediction interval this observation is $(19.504, 39.863)$.