Question 1

Alvee Chowdhury

28/01/2021

Set up:

```
set.seed(42)
h1q1 <- read.csv("h1q1.csv")
train_index = sample(1:nrow(h1q1), size = round(0.5 * nrow(h1q1)))
train_data = h1q1[train_index, ]
test_data = h1q1[-train_index, ]
```

fitting the four models:

```
Model_1 <- lm(y ~ .,data=train_data)
Model_2 <- lm(y ~ . + I(a^2) + I(b^2) + I(c^2), data= train_data)
Model_3 <- lm(y ~ . ^ 2 + I(a^2) + I(b^2) + I(c^2), data=train_data)
Model_4 <- lm( y ~ a * b * c * d + I(a^2) + I(b^2) + I(c^2), data= train_data)
```

Defining RMSE function and complexity function:

```
# Calculates RMSE

rmse = function (actual,predicted){
  sqrt(mean((actual -  predicted)^2))
}

get_rmse = function (model,data, response){
  rmse (actual = subset(data, select = response, drop = T),
        predicted = predict(model,data))
}

# Calculates the number of parameters, excluding the variance parameter
# of the model:
parameter_number = function (model) {
  length(coef(model)) - 1
}
```

Now let us obtain the train RMSE,test RMSE and number of parameters (excluding variance parameter) for each model defined above:

```
# (a) train RMSE
model_1_train_RMSE <- get_rmse (model = Model_1, data = train_data, response = 'y')
model_2_train_RMSE <- get_rmse (model = Model_2, data = train_data, response = 'y')
model_3_train_RMSE <- get_rmse (model = Model_3, data = train_data, response = 'y')
model_4_train_RMSE <- get_rmse (model = Model_4, data = train_data, response = 'y')

# (b) test RMSE
model_1_test_RMSE <- get_rmse (model = Model_1, data = test_data, response = 'y')
```

```r
model_2_test_RMSE <- get_rmse (model = Model_2, data = test_data, response = 'y')
model_3_test_RMSE <- get_rmse (model = Model_3, data = test_data, response = 'y')
model_4_test_RMSE <- get_rmse (model = Model_4, data = test_data, response = 'y')

# (c) the number of parameters, excluding the variance parameter:
Model_1_number <- parameter_number(Model_1)
Model_2_number <- parameter_number(Model_2)
Model_3_number <- parameter_number(Model_3)
Model_4_number <- parameter_number(Model_4)
```

```r
library(flextable)
table <- data.frame(
  Model = c("Model 1", "Model 2", "Model 3", "Model 4"),
  model_train_RMSE = c(model_1_train_RMSE, model_2_train_RMSE, model_3_train_RMSE, model_4_train_RMSE),
  model_test_RMSE = c(model_1_test_RMSE, model_2_test_RMSE, model_3_test_RMSE, model_4_test_RMSE),
  model_number = c (Model_1_number,Model_2_number,Model_3_number,Model_4_number))


t<-delete_part(flextable(table), part = "header")
t<- add_header(t,top=T,model_train_RMSE="model train RMS",model_test_RMSE= 'model test RMSE',model_numb



m<- colformat_num(t,j=c(2,3),digits = 7)
j<- colformat_num(m,j=c(4),digits = 0)



autofit(align(j,align = "center", part = "all"))
```

	model train RMS	model test RMSE	number of parameters
Model 1	1.4483400	1.4247811	4
Model 2	1.1596770	1.1197802	7
Model 3	0.5070891	0.5269903	13
Model 4	0.5044712	0.5332113	18

We see that 'Model 1' is the least flexible model, and 'Model 4' is the most flexible model.
Moreover, we see that the Train RMSE decrease as flexibility increases. The Test RMSE is smallest for 'Model 3', so it is the model we believe to perform the best on data not used to train the model.
In general, 'Model 1' and 'Model 2' have high values of Train RMSE and high values of Test RMSE. Hence underfitting.
Model 4 seem to have overfitting.
To conclude, Model 3 is the best model out of these 4.