Homework 02

STAT 430, Fall 2017

Due: Friday, September 22, 11:59 PM

Please see the homework instructions document for detailed instructions and some grading notes. Failure to follow instructions will result in point reductions.

Exercise 1

[**15 points**] This exercise will use data in `hw02-train-data.csv` and `hw02-test-data.csv` which are train and test datasets respectively. Both datasets contain a single predictor `x` and a numeric response `y`.

Fit a total of 20 linear models. Each will be a polynomial model. Use degrees from 1 to 20. So, the smallest model you fit will be:

- `y ~ poly(x, degree = 1)`

The largest model you fit will be:

- `y ~ poly(x, degree = 20)`

For each model, calculate Train and Test RMSE. Summarize these results using a single plot which displays RMSE (both Train and Test) as a function of the degree of polynomial used. (Be sure to make the plot easy-to-read, and well labeled.) Note which polynomial degree appears to perform the "best," as well as which polynomial degrees appear to be underfitting and overfitting.

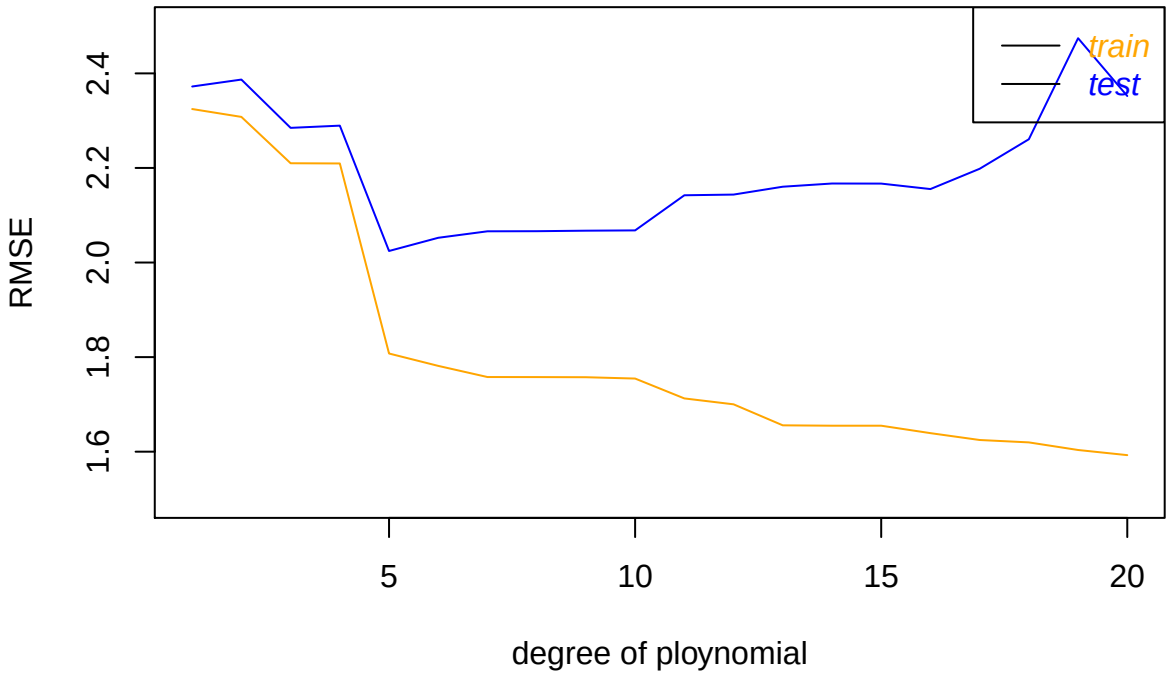
```
rmse = function(actual, predicted) {
  sqrt(mean((actual - predicted) ^ 2))
}
get_rmse = function(model, data) {
  rmse(actual = data[, 'y'],
       predicted = predict(model, data))
}
polymodel = function(data,d){
   lm(y ~ poly(x, degree = d), data = data)
}
result = function(train,test,d)
{
  model = polymodel(train,d)
  get_rmse(model,test)
}
```

```
train_data = read.csv("hw02-train-data.csv")
test_data = read.csv("hw02-test-data.csv")
```

```
d = 1:20
test_rmse = sapply(d,result,train = train_data,test = test_data)
train_rmse = sapply(d,result,train = train_data,test = train_data)
```

```
df = data.frame(train_RMSE = train_rmse, test_RMSE = test_rmse, degree = 1:20)
#df
plot(df$degree,df$test_RMSE,type = "l",col = "blue",xlab = "degree of ploynomial",ylab = "RMSE",ylim = c
```

```
lines(df$degree,df$train_RMSE,col = "orange")
legend("topright",text.col = c("orange","blue"),legend = c("train","test"),cex = 1, lwd = 1, text.font =
```




```
Before the degree of 5, models are underfitting, model is best when degree equals to 5, and all models
```

Exercise 2

[15 points] This exercise will again use data in `hw02-train-data.csv` and `hw02-test-data.csv` which are train and test datasets respectively. Both datasets contain a single predictor `x` and a numeric response `y`.

Fit a total of 10 nearest neighbors models. Each will use a different value of `k`, the tuning parameter for the number of neighbors. Use the values of `k` defined in the following `R` chunk.

```
k = seq(5, 50, by = 5)
train_x = train_data["x"]
test_x = test_data["x"]
train_y = train_data$y
test_y = test_data$y
```

When fitting the models, scale the `x` variable using the `scale()` function. Use the default arguments for `scale()`. That is, do not supply any arguments beyond the data to be scaled.

```
get_rmse_knn = function(x_tst, y_tst, k) {
  pred = knn.reg(train = train_x, test = x_tst,
                 y = train_y,k = k)$pred
  rmse(actual = y_tst, predicted = pred)
}
```

```
library(FNN)
train = sapply(k,get_rmse_knn,x_tst = train_x,y_tst = train_y)
test = sapply(k,get_rmse_knn,x_tst = test_x,y_tst = test_y)
```

For each value of the tuning parameter, calculate Train and Test RMSE. Summarize these results using a single well-formatted table which displays RMSE (both Train and Test), `k`, and whether or not that value of the tuning parameter appears to be overfitting, underfitting, or the "best" value of the tuning parameter. Consider rounding your results to show only enough precision to choose the "best" model.

```
best_k = k[which.min(test)]
fit_status = ifelse(k < best_k, "Over", ifelse(k == best_k, "Best", "Under"))

result = data.frame(Train_RMSE = round(train,2),
                    Test_MSE = round(test,2),
                    K = k,fitting = fit_status)
knitr::kable(result)
```

Train_RMSE	Test_MSE	K	fitting
1.65	2.16	5	Over
1.70	2.08	10	Over
1.79	2.05	15	Best
1.93	2.06	20	Under
2.02	2.14	25	Under
2.28	2.36	30	Under
2.60	2.67	35	Under
2.96	2.99	40	Under
3.27	3.29	45	Under
3.58	3.57	50	Under