Team 2 - Homework Two

Assignment 2: KJ 7.2; KJ 7.5

NAME

DATE

Dependencies

```r
# predictive modeling
libraries('mlbench', 'caret')

# Formatting Libraries
libraries('default', 'knitr', 'kableExtra')

# Plotting Libraries
libraries('ggplot2', 'grid', 'ggfortify')
```

(1) Kuhn & Johnson 7.2

> Friedman (1991) introduced several benchmark data sets create by simulation. One of these simulations used the following nonlinear equation to create data: $y = 10\sin(\pi x_1 x_2) + 20(x_3 - 0.5)^2 + 10x_4 + 5x_5 + N(0, \sigma^2)$; where the x values are random variables uniformly distributed between $[0, 1]$ (there are also 5 other non-informative variables also created in the simulation).

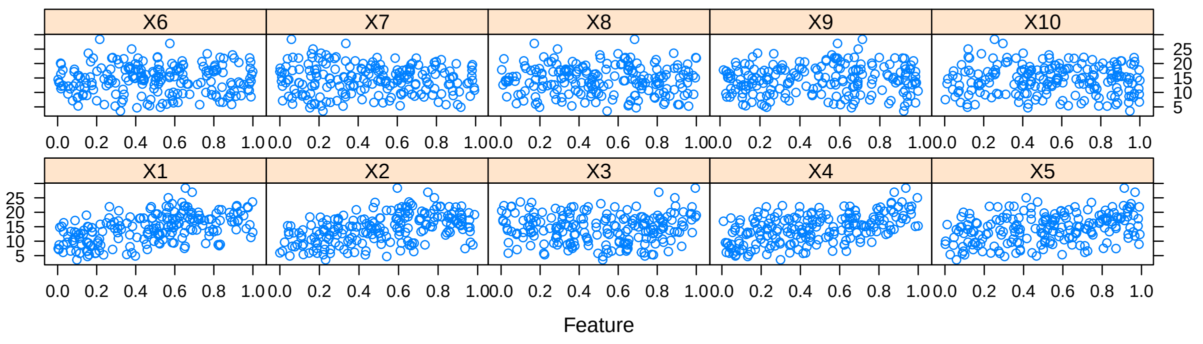
The package `mlbench` contains a function called `mlbench.friedman1` that simulates these data:

```r
set.seed(200)
trainingData <- mlbench.friedman1(200, sd = 1)

## We convert the 'x' data from a matrix to a data frame
## One reason is that this will give the columns names.

trainingData$x <- data.frame(trainingData$x)

## Look at the data using
featurePlot(trainingData$x, trainingData$y)
```



```r
## or other methods.

## This creates a list with a vector 'y' and a matrix
## of predictors 'x'. Also simulate a large test set to
## estimate the true error rate with good precision:

testData <- mlbench.friedman1(5000, sd = 1)
testData$x <- data.frame(testData$x)
```

(a) Tune several models on these data. For example:

```r
knnModel <- train(x = trainingData$x,
                  y = trainingData$y,
                  method = "knn",
                  preProc = c("center", "scale"),
                  tuneLength = 10)
knnModel
```

```
k-Nearest Neighbors

200 samples
 10 predictor

Pre-processing: centered (10), scaled (10)
Resampling: Bootstrapped (25 reps)
Summary of sample sizes: 200, 200, 200, 200, 200, 200, ...
Resampling results across tuning parameters:

  k   RMSE      Rsquared   MAE
   5  3.565620  0.4887976  2.886629
   7  3.422420  0.5300524  2.752964
   9  3.368072  0.5536927  2.715310
  11  3.323010  0.5779056  2.669375
  13  3.275835  0.6030846  2.628663
  15  3.261864  0.6163510  2.621192
  17  3.261973  0.6267032  2.616956
  19  3.286299  0.6281075  2.640585
  21  3.280950  0.6390386  2.643807
  23  3.292397  0.6440392  2.656080

RMSE was used to select the optimal model using the smallest value.
The final value used for the model was k = 15.
```

```r
knnPred <- predict(knnModel, newdata = testData$x)

## The function 'postResample' can be used to get the test set performance values
postResample(pred = knnPred, obs = testData$y)
```

```
     RMSE  Rsquared       MAE
3.1750657 0.6785946 2.5443169
```

Model 1:

```
#code
```

Model 2:

```
#code
```

Model 3:

```
#code
```

(b) Which models appear to give the best performance? Does MARS select the informative predictors (those named X1-X5)?

```
#code
```

(2) Kuhn & Johnson 7.5

Exercise 6.3 describes data for a chemical manufacturing process. Use the same data imputation, data splitting, and pre-processing steps as before and train several nonlinear regression models.

```
# Call code from 6.3
```

(a) Which nonlinear regression model gives the optimal resampling and test set performance?

```
# code
```

(b) Which predictors are most important in the optimal nonlinear regression model? Do either the biological or process variables dominate the list? How do the top ten important predictors compare to the top ten predictors from the optimal linear model?

```
# code
```

(c) Explore the relationships between the top predictors and the response for the predictors that are unique to the optimal nonlinear regression model. Do these plots reveal intuition about the biological or process predictors and their relationship with yield?

```
# code
```