**Computational Statistics II - Homework 2**

Subash Kharel

Southern Illinois University, Carbondale

1. **Answer:**
   1. **When the sample size *n* is extremely large, and number of predictors p is small.**  
      *Since the size of n is extremely large, we won’t have problem of overfitting the data using flexible model. Thus, we can use flexible model here.*
   2. **The number of predictors p is extremely large, and the number of observations n is small.**  
      *With the small number of observations, we need to use inflexible model since it prevents overfitting of data rather than flexible model that will, for sure, overfit the data.*
   3. **The relationship between the predictors and response is highly non-linear.***Flexible model will be the best fit here since this model can be used to fit the non-linear data.*
   4. **The variance of the error terms, i.e. σ2 = Var(ε), is extremely high.**  
      *Since the variance of error is large, employing the flexible model will capture a lot of noise and degrades the performance. Inflexible model is the best in this scenario.*
2. **Answer for 7:**Firstly, the data is converted into data frame using the code below:  
     
   *#Creating data frame to perform data operations*

*x1 <- c(0, 2, 0, 0, -1, 1)*

*x2 <- c(3, 0, 1, 1, 0, 1)*

*x3 <- c(0, 0, 3, 2, 1, 1)*

*y <- c("Red", "Red", "Red", "Green", "Green", "Red")*

*y <- as.factor(y)*

*data <- data.frame(x1, x2, x3, y)*

* 1. Code:  
       
     *#Calculating eucledian distance for (0, 0, 0)*

*p <- c(0, 0, 0)*

*distance <- rep(0, 6)*

*for(i in 1:6) {*

*q <- c(x1[i], x2[i], x3[i])*

*z <- rbind(p, q);*

*distance[i] <- dist(z, method = "euclidean")*

*}*

*#Adding distance vector to data frame and sorting the data frame by the distance column*

*data$Distance <- distance*

*data <- data[order(data$Distance),]*  
  
Output:  


* 1. Code  
     #Function to estimate mode  
     Copied from : https://stackoverflow.com/questions/2547402/is-there-a-built-in-function-for-finding-the-mode#answer-8189441  
     estimate\_mode <- function(x) {

ux <- unique(x)

ux[which.max(tabulate(match(x, ux)))]

}  
#Using K= 1 for prediction

prediction = estimate\_mode(data$y[1:1])

message("--->With K=1, the prediction is ", as.character(prediction))  
  
Output:  
--->With K=1, the prediction is Green

* 1. Code  
     #Using K= 3 for prediction

prediction = estimate\_mode(data$y[1:1])

message("--->With K=3, the prediction is ", as.character(prediction))

Output:  
--->With K=3, the prediction is Red

* 1. *If Bayes decision boundary in the problem is highly non-linear, we can use different value of K. Using small value of K makes the model very flexible so that it captures the training data well but may fail in classifying the test data. Using large value of K makes the model inflexible and may give some percentage of error on the training data but, will classify the test data with some degree of error.  
     Thus, the choice of K depends upon the flexibility requirement of the model. However, in my case I will go with the large value of K.*