Assignment 1: Some Basic Ideas in Machine Learning

BUAD 5072 – Fall 2016

---

1. Objectives

The purpose of this assignment is to provide you with an opportunity to investigate some of the ideas regarding the assessment of model accuracy that we have been discussing in class.

2. What You Will Need

- Access to a Windows computer with R, and to the following files, which can be downloaded from the Class Schedule page of the course web site:
  - HomePrices.txt
  - LoadData.csv

3. What You Will Hand In

Submit your script file as Assignment1.R via Blackboard - Assignment 1.

4. Due Date

Friday October 21st just before midnight.

5. Note on Collaboration

This is a Category A assignment. Specifically, you may not receive help from anyone on this assignment except the professor or teaching assistant. It must be 100% your own work. All questions concerning this assignment must be addressed to your professor or the teaching assistant. **It is an honor code offense to give or receive any assistance on these assignments**.

6. Preliminaries:

To get set up for the assignment, follow these steps:

1. As the first statement in your script file, enter rm(list=ls())
2. Each question in the assignment should begin with the following three comment lines, where *n* is the question number:

   ```
   ######################
   #### QUESTION n ####
   ######################
   ```

3. I should be able to run your script on my computer without errors or interruptions. For this to happen, you must:
   a. Avoid entering file path information…my files will be located in a different location that yours, and so your code will fail on my machine. Instead, always refer only to files in your working directory.
   b. Do not use functions like file.choose(), fix(),edit(), or q()
4. Do not create console output other than what is asked of you explicitly. For example, in your final script, remove any statements that you used to verify the contents or structure of data.
5. I suggest that you read the entire assignment before starting – there are often notes and suggestions at the end of the assignment document.

7. Assignment Tasks:

Part 1: Predicting House Prices: The Regression Setting (50%)

- Create a data frame from the file named HomePrices.txt. which contains median home values for 506 neighborhoods in and around Boston. In the following steps, you will use this dataset to construct a series of models to predict this median home value (the medv variable) using all the other variables as predictors. Do the following:
- Print the MSE that would result from always predicting the mean of medv
- Print the Variance of medv, assuming that the data represent a population, not a sample
  - Hint: Use the var() function and then multiply by (n-1)/n, where n is the total number of observations in the full dataset
- Set seed to 5072
- Create training and test sets – training set should have .8*n observations
- Using the knn.reg() function (part of the FNN package), predict the median value of homes (the medv variable) using all the other variables as predictors.

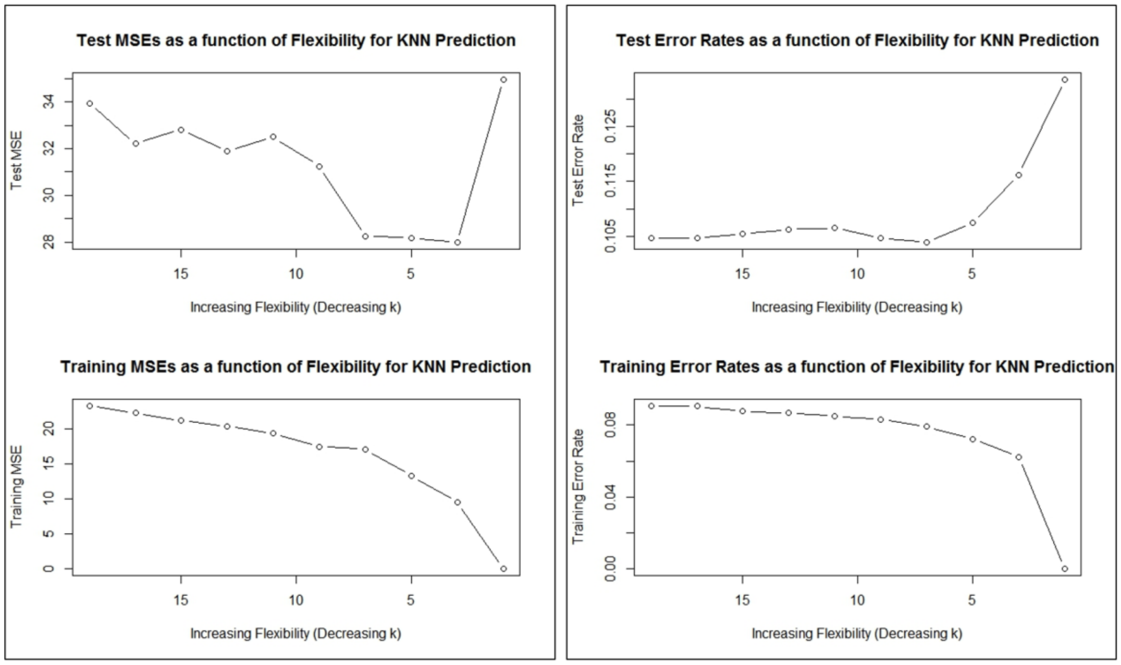
    - Because the “nearest-neighbor” models you will build are based on Euclidean distance, scale is important. You will need to scale and center the predictors (but not the variable you are predicting – we will preserve its scale for comparison with the MSE computed above)
    - Construct 10 models with k equal to 19, 17, 15, …, 1, evaluating both the test and training MSE’s for each k.
        - Note that for each k, you will need to execute the knn.reg() function twice, once to predict the training dependent variables and once to predict the test dependent variables in order to compute the MSE’s for each.
    - The knn.reg() function requires 4 input parameters and produces a named list as output. The list element named $pred contains the predictions. The input parameters are:
        - A matrix or data frame containing the predictors associated with the training data
        - A matrix or data frame containing the predictors associated with the data for which we wish to make predictions
        - A factor containing the class labels for the training observations
        - A value for K, the number of nearest neighbors to be used by the classifier.
- Plot the training and test MSE’s as a function of the k’s. The x-axis should go from least flexible to most flexible – left to right.
- Print the k and associated MSE that produced the lowest training MSE. Do the same for the test MSE.

<u>Part 2</u>: Predicting Loan Repayment: The Classification Setting (50%)

- Create a data frame from the file named LoanData.csv. which contains data on bank loans. Using this dataset, you will construct a series of models to predict whether or not borrowers will repay their loans (the loan_repaid variable) using all the other variables as predictors.
- Print the error rate that would result from always predicting Yes
- Set seed to 5072
- Create training and test sets – training set should have .8*n observations, where n is the total number of observations in the full dataset
- Using the knn() function (part of the class package), predict the loan repayment (the loan_repaid variable) using all the other variables as predictors.
    - Because the “nearest-neighbor” models you will build are based on Euclidean distance, scale is important. You will need to scale and center the predictors (but not the variable you are predicting, which is categorical)
    - Construct 10 models with k equal to 19, 17, 15, …, 1, evaluating both the test error rates and training error rates for each k.
        - Note that for each k, you will need to execute the knn() function twice, once to predict the training dependent variables and once to predict the test dependent variables in order to compute the error rates for each.
    - The knn () function requires 4 parameters and produces a factor of predictions. The parameters are:

    - A matrix or data frame containing the predictors associated with the training data
    - A matrix or data frame containing the predictors associated with the data for which we wish to make predictions
    - A factor containing the class labels for the training observations
    - A value for K, the number of nearest neighbors to be used by the classifier.
- Plot the training and test error rates as a function of the k's. The x-axis should go from least flexible to most flexible – left to right.
- Print the k and associated error rate that produced the lowest training error rate. Do the same for the test error rates.

Your results for parts1 and 2 should look something like this:



Some additional notes:

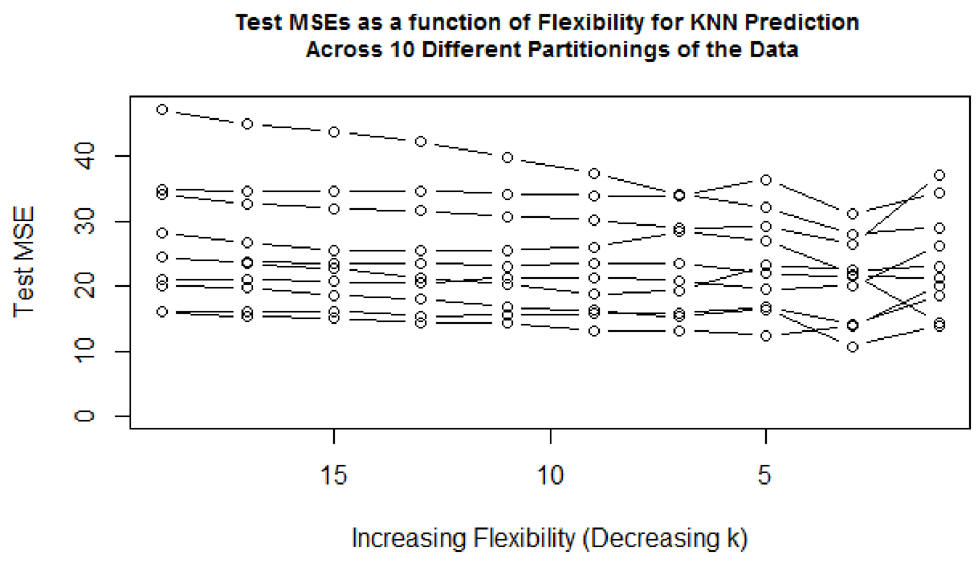
The training set X's are used to construct a model that we hope will predict a correct y when shown a non-training X. But the model we construct from the training set X's won't be able to perfectly predict the training set Y's – it will make mistakes even on the observations used to create it. These errors are used to compute the training MSE.

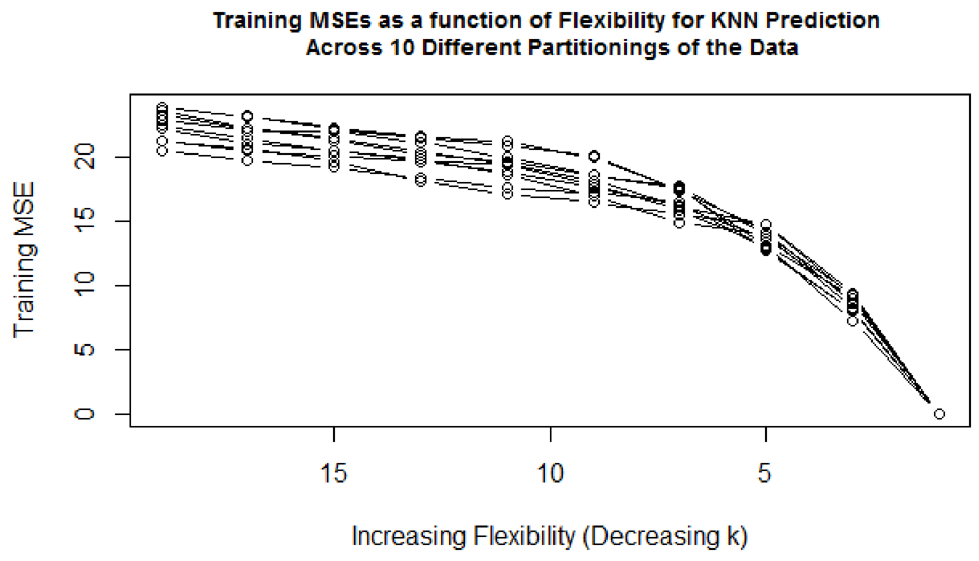
In one call of the knn.reg() function, we will give it the following parameters: the training X's, the training X's again, and the training Y's, and k. It will then create a model using the (first parameter) training X's and the (third parameter) training Y's, then use the same X's (the second parameter) to predict corresponding Y's. By comparing these predicted Y's with the actual training Y's, we can compute the training MSE.

To produce the test MSE, we will again call the knn.reg() function, but this time replacing the second parameter with the test X's. It will create the same model as above, but will use the test X's to predict their corresponding Y's. These X's have not been "seen" by the algorithm creating the model, so we would expect that the error rate would be higher than above. By comparing the predicted Y's to the actual test Y's (which we did not use in either of the above calls to knn.reg() function), we can compute the test MSE.

Part 3: (Optional – 10% Extra Credit) Investigating the Variance of the knn.reg model on the home pricing dataset used in Part 1:

- Repeat Part 1 with 10 different random partitions of the data into training and test sets, and plot the resulting training and test MSE's. With the seed set to 123, I obtain the following plots:


Test MSEs as a function of Flexibility for KNN Prediction
Across 10 Different Partitionings of the Data
Test MSE
Increasing Flexibility (Decreasing k)


Training MSEs as a function of Flexibility for KNN Prediction
Across 10 Different Partitionings of the Data
Training MSE
Increasing Flexibility (Decreasing k)