Predicting Heart Disease with KNN

Scott Schirkofsky

MSDS 680 Machine Learning

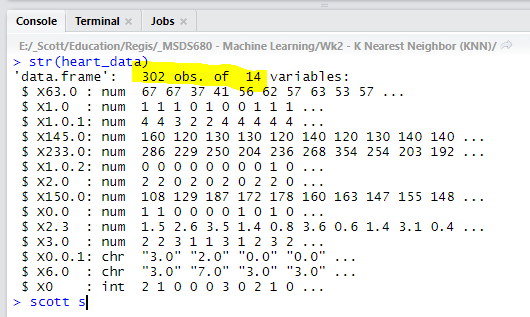
Regis University

Predicting Heart Disease with KNN

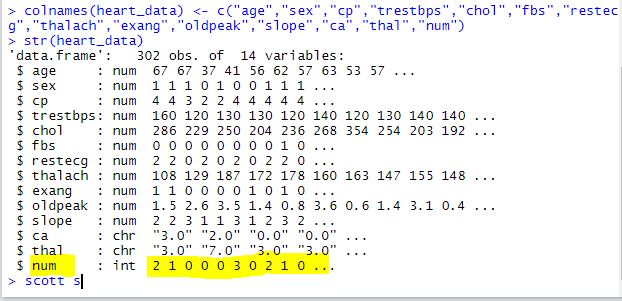
**Objective:** to predict heart disease in patients. Using https://archive.ics.uci.edu/ml/datasets/Heart+Disease #use "processed.cleveland.data".

* 1. **Get to know your data, start out by data exploration (show the results with the corresponding discussion/interpretation). Step 1 & 2 collecting and exploring the data.**

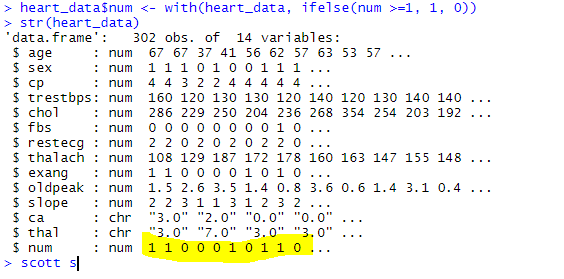
Using str(heart\_data) shows that the dataset has 302 examples and 14 attributes. The dataset does not contain a unique identifier id that would need to be excluded.



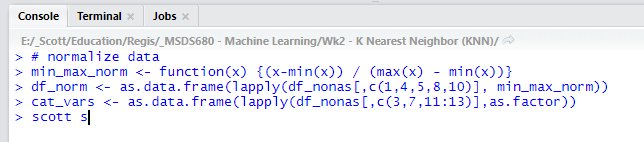
It is also apparent that the attributes have no column names so those will need to be added using the attribute information provided for this dataset. In addition, the documentation for this dataset shows that the “num” feature represents the diagnosis of heart disease which is also what I am trying to predict in this assignment. When num is equal 0 the observation is without heart disease, when num is greater than or equal to 1 there is heart disease.



Next the values for num can converted to 1 when the value is 1 or greater to simplify things.



Next the data will be normalized to aid with model’s performance. Here I use “min”, “max”, and “apply” to to find how far each value is from the min and max. The results are then applied by using lapply.



* 1. **Divide the data into a training set and a test set randomly with ratio 70:30. Make the prediction based on 1-nearest neighbor. What is the error rate of this approach? Report the confusion matrix and accuracy. Interpret the results.**

The classifications of interest are the of the “true” variety TP and TN. 50 true positives were predicted along with 37 true negatives (mistakes). This means that there were 50 true positives for heart disease and 37 true negatives or correctly rejected. Using the confusion matrix data, the error rate and accuracy are calculated below. There were 2 FP false alarms (type 1 error), and 5 FN misses (type 2 error).

**Error rate:** (FP + FN) / (TP + TN + FP + FN)

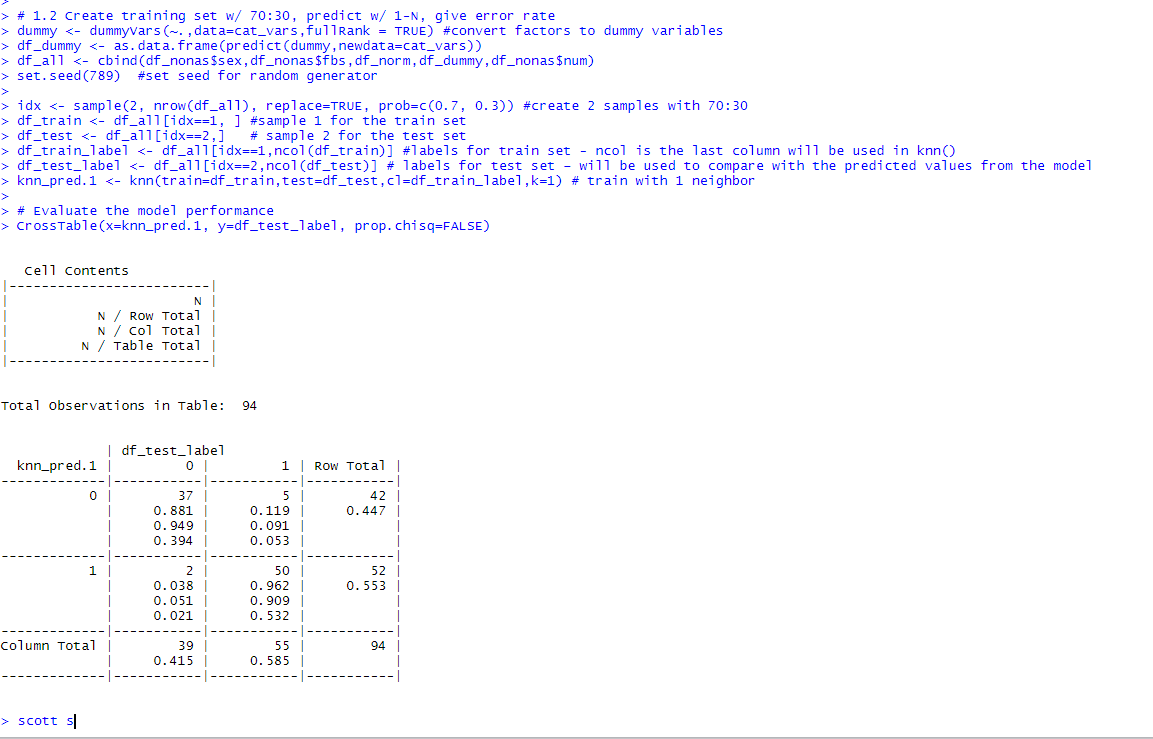
> (2 + 5) / (50 + 37 + 2 + 5)  
 > 0.0745

**Accuracy:** (TP + TN) / (TP + TN + FP + FN)  
 > (50 + 37) / (50 + 37 + 2 + 5)  
 > 0.9255

**True Positive:** 50  
**True Negative:** 37

**False Positive:** 2 (Type 1 error)

**False Negative:** 5 (Type 2 error)



* 1. **Use different values for K, what is the optimal value of K from your experiments?**

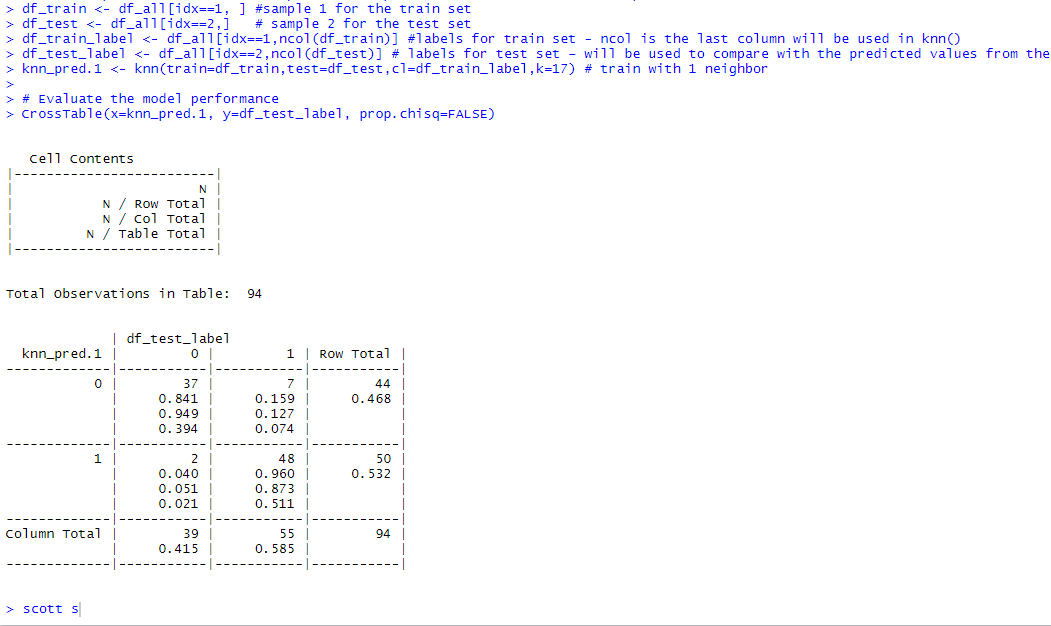
The optimal value for k with 302 examples is 17 because 17 is the square root of 302 and is also an odd number. The odd number will help eliminate ties.

**Report the error rate of the optimal K value and its confusion matrix. Is there any improvement (by how much) over 1-nearest neighbor? You can fast forward to week 7 FTE to learn more about performance evaluation. Perhaps, these questions may pop up into your minds:**

The error rate actually went up from .0745 to .0957 when k was increased to its optimal value of 17 so there was no improvement with the increase. However, the number of mistakes stayed the same at 37.

**Error rate:** (FP + FN) / (TP + TN + FP + FN)

> (7 + 2) / (48 + 37 + 7 + 2)  
 > .0957



**What library should be included for this assignment?**

Three additional libraries are needed for this assignment. The “caret” library is needed for dummyVars in order to create dummy variables. The “class” library is also needed for the kNN function used to train the data, and gmodels was the third library needed for the CrossTable function that creates a confusion matrix.

**How do you clean up the input data?**

To clean the data the NA values were removed, num values were converted to 0 and 1, and the factors were normalized. 5 numeric columns were normalized with a min/max function while 5 vectors were converted to factors.

**Should we normalize the data?**

Yes, and it should be done after NA values are removed so that the NAs do not influence the normalization of factors.