Alex Te (861215867)
CS 171 Spring 2018
Professor Papalexakis
Late Days used for this assignment: 0
Total Late Days used so far: 1

Question 0: Getting real data

I first imported data from UCI's Machine Learning Repository, specifially the Breast Cancer Wisconsin (Original) Data Set. I did so by creating a script in python that would read in the data, store it into an array, and parse it. I also handled missing data by replacing the "?" values with 0 so that it would make computation easier.

Question 1: k-Nearest Neighbor Classifier

To implement the knn algorithm, I first took 699 data points and split 80% of it for training data and 20% for testing data. Then I pass in both the smaller datasets into my knn algorithm where it would compute the distance from a single testing data (from the testing dataset as a whole) to all the training data. It would then choose the k nearest values (nearest neighbors) and return the class that it belongs to.

How I handled a tie:

If the testing data is checking for 1 neighbor, then whatever that neighbor's class is, the testing will also be of that class. However, if there are 2 neighbors and each are different (2 and 4) either one will work. Else, the class is determined by the majority vote.

To calculate the distance, I used the Lp norm distance calculation which is the summation of ((x_i - y_i) ^ p) ^(1/p) for i = 1, 2, 3, . . ., n

We pass in x_test (which is the 20% of the data, or testing data), x_train (which is the training data, 80%), y_train (which has 1-1 corresponding number of items, each of which is the x_train's class label), k which is the # of nearest neighbors, and p which is the parameter of Lp distance. What is returned from the knn function is a list of the classifier's results of the testing set. I take the returned list and compare it with a variable called y_test which holds the actual class's prediction. Then I take the % right and divide it by the total to get the accuracy which in the case of k = p = 1 yields 99.28%.

Question 2: Evaluation

1. Cross-Validation (code is attached)
    a. First I randomized that 699 data.
    b. To implement the 10-fold cross-validation, I first made 10 equal size "folds". However, 699 / 10 = 69.9 so one fold had 69 which the 9 others had 70
    c. Then we pass one fold as the testing and the other 9 as the training. After passing the first fold, pass the next fold and the other 9 as training. Repeat until all the folds have been a testing fold once.

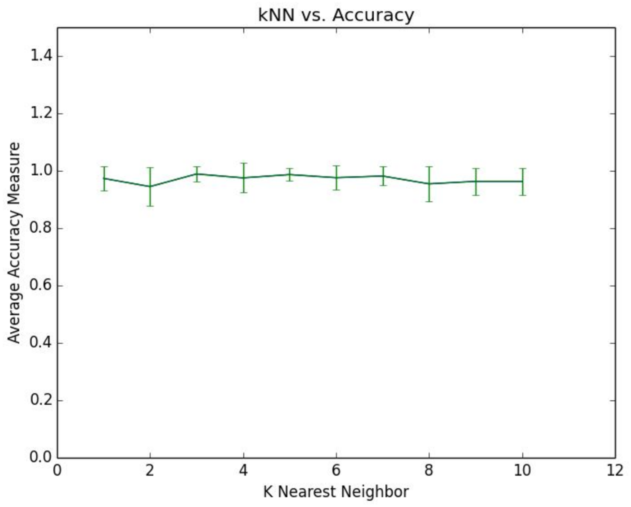
i. After each fold calculation, find the accuracy, sensitivity, and specificity and store it into a list because we want to take the average of each
ii. Graph the average accuracy, specificity, and sensitivity and graph them knn vs their averages.

d. **ASSUMPTION**: I had 4 as my positive class and 2 as my negative class.

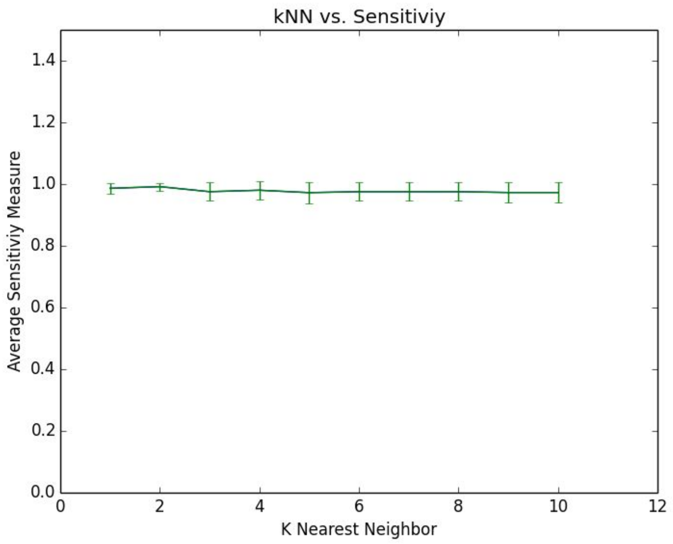
2. Evaluating knn. These graphs are the the representation for data with average accuracy, sensitivity, and specificity. First 3 are for p = 1, and the next 3 are k = 2.

P = 1

KNN VS ACC



KNN VS SENSITIVITY



KNN VS SPECIFICITY

P = 2

KNN VS ACCURACY

KNN VS SENSITIVITY

KNN VS SPECIFICITY

The combination that yields the best result: K = 2, P = 1. The error bar is small and also the average sensitivity is near 100%