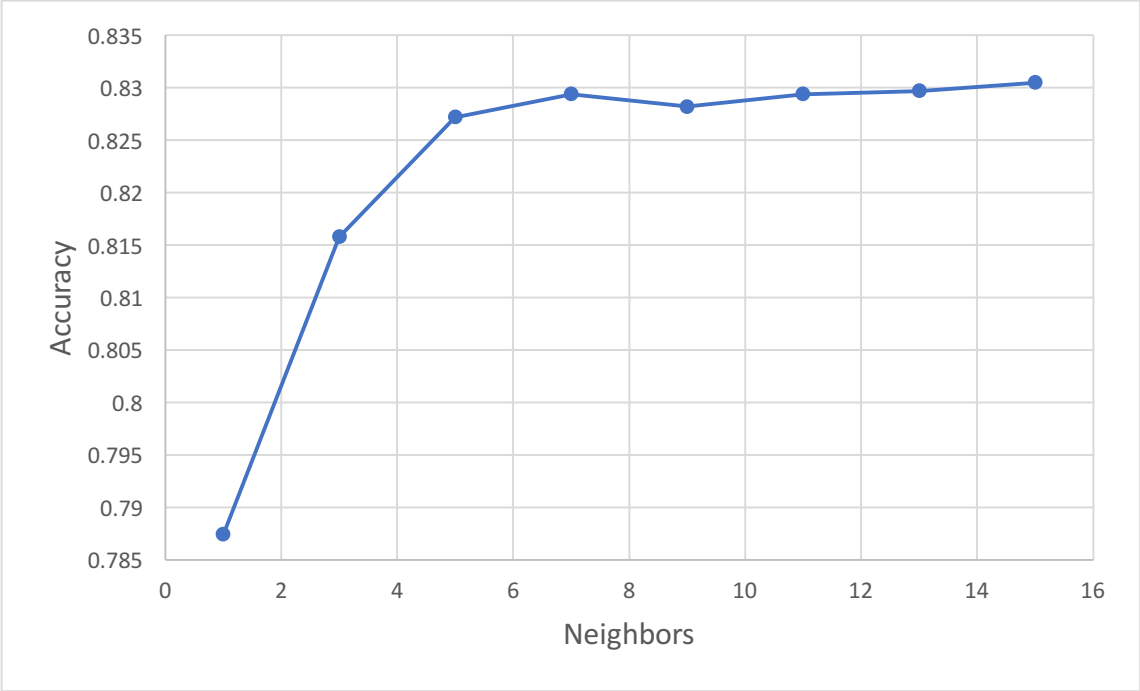
Magic Telescope (Non-Weighted)

Normalization:

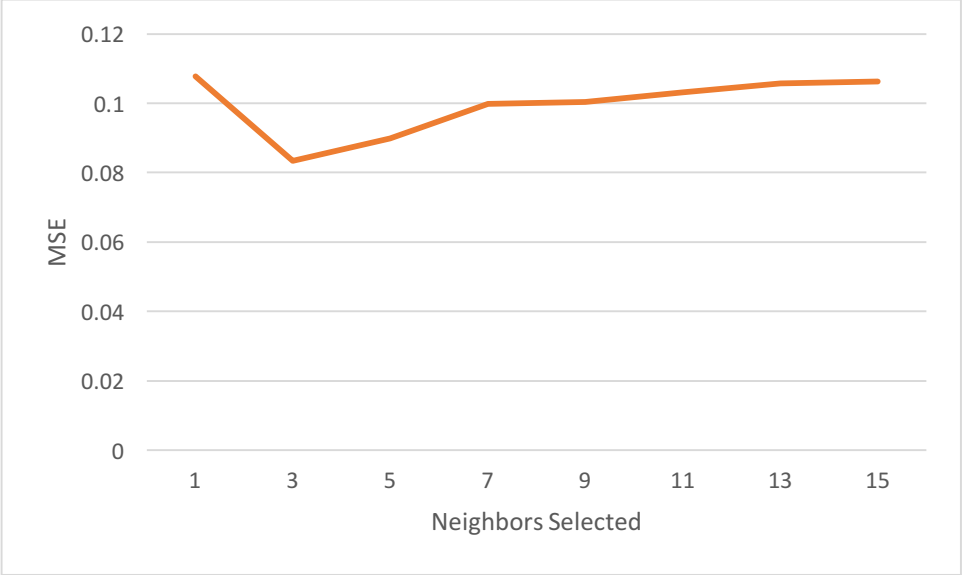
Running with 3 nearest neighbors with data normalized was getting me about 81.5% accuracy while if I ran it non-normalized it was getting me about 80% accuracy. So, normalizing the data definitely seems like it's worth it to get that extra accuracy. Normalizing helps reduce misclassification as it helps make the Euclidean distance formula more scaled and thus more accurate. This normalization essentially helps weight differences more equally and gives helps all labels have an impact in determining Euclidean distance.

As you can see in the chart below it seems that k value 15 has the most accurate value and barely squeaks out the others. However odd values 5 through 13 seem to be roughly the same accuracy and value 5 took a lot less time to run than 13, so I would say that 5 nearest neighbors is the best for the magic telescope data set due to it being able to run faster.



KNN Regression Housing Price Prediction (Non Weighted) MSE

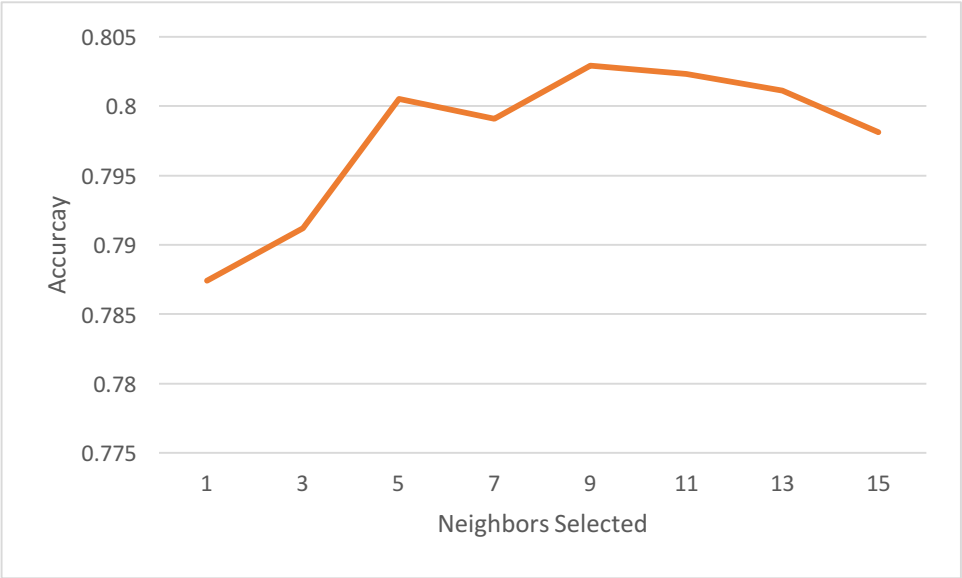
As you can see in the chart below for this data set it appears that 3 nearest neighbors works the best and produces the lowest means squared error of all the odd values 1 through 15. I used the regression algorithm to calculate the mean squared error.



Weights

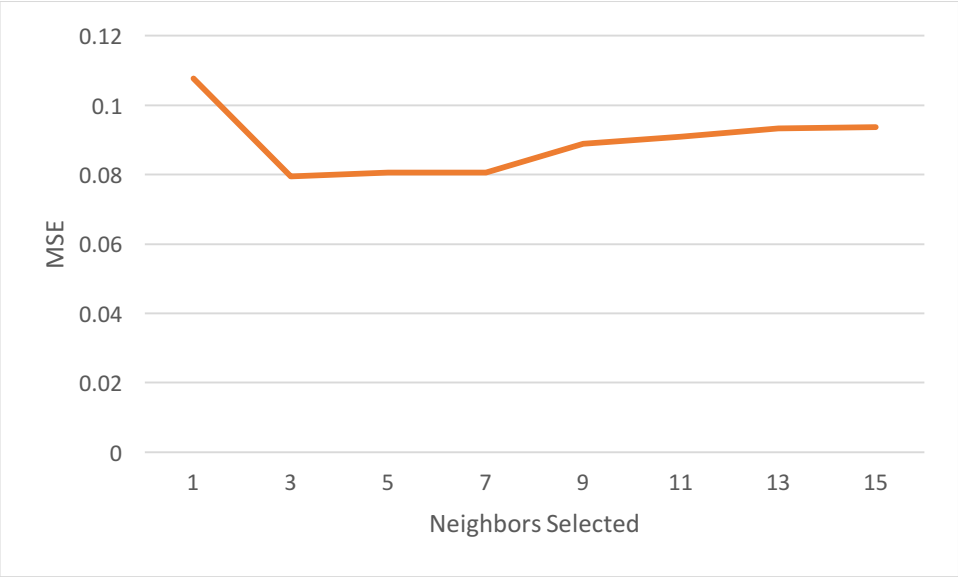
Magic Telescope Weighted

Interestingly the weighted values on the magic telescope seemed to be slightly less accurate then the unweighted values (although not by much). They also seemed to be a lot more resistant to change when the number of neighbors is increased. The range from 1 to 15 neighbors for weighted values was only from 0.787 to 0.803, while with unweighted it was from 0.787 to 0.831. To me, it would make sense that weighted neighbors wouldn't change as much because it would be more accurate from the get from using this weighted formula and would be more precise in weighting the influence of certain neighbors.



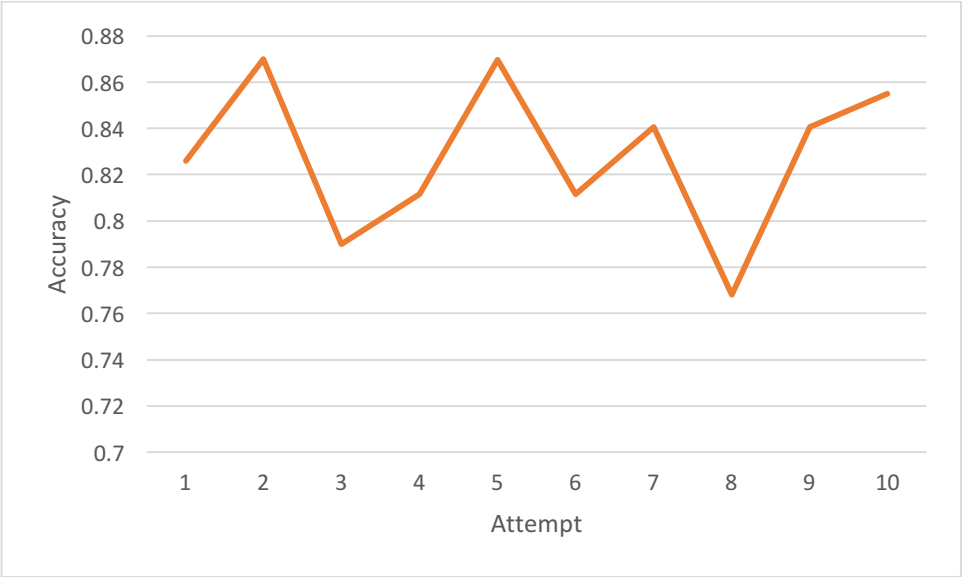
KNN Regression Housing Price Prediction(Weighted) MSE

A**s** you can see in the graph below running our weighted algorithm with MSE has a slight improvement as compared to the non-weighted attempt and seemed to have about one Percent improvement across the board. It again seems that three neighbors is our best bet to run this data set on.



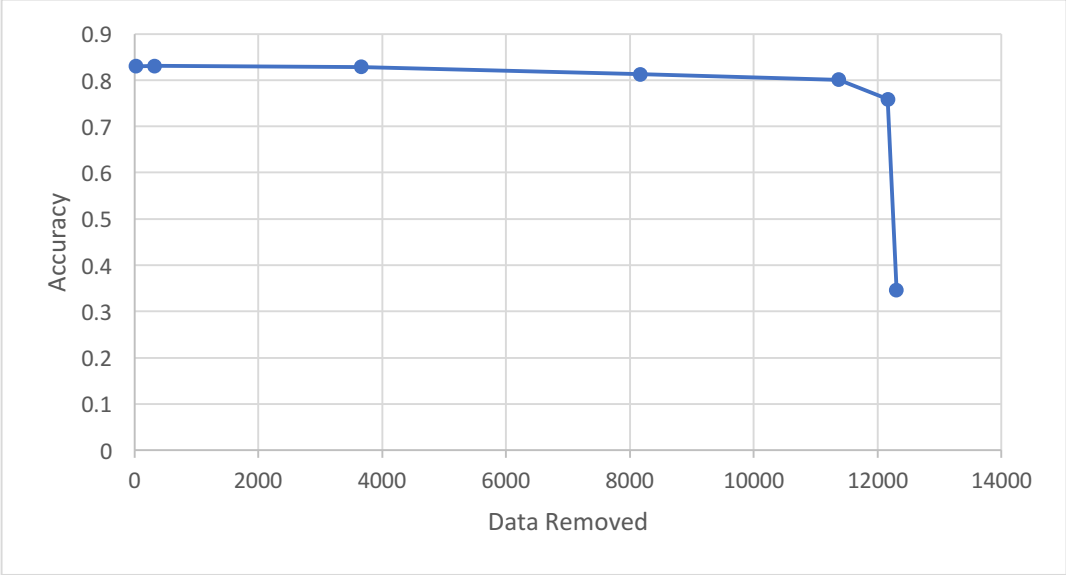
Credit Approval

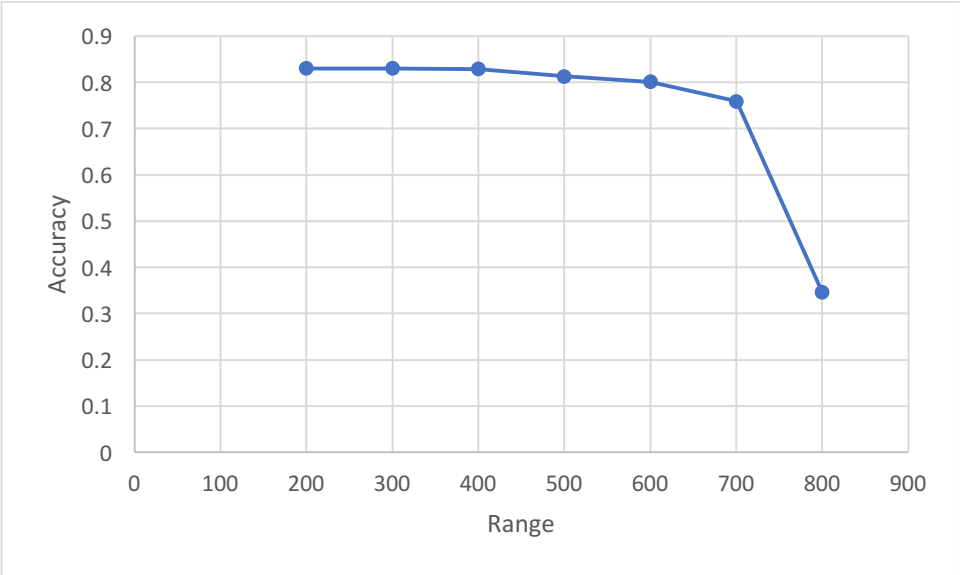
For missing data in this problem, I simply modified by distance calculating function to have a distance of 1 when comparing that missing data. For nominal attributes, I would compare to see if the two objects had the same type. If they did, then their distance would be set to zero for those attributes, otherwise It would be set to one. I believe that this is a fair approach as we really can' just arbitrarily assign random values to it as that wouldn't be accurate as we wouldn't know what values to choose. As you can see below in the graph all accuracies are in the 76 to 87 range. I ran with 15 neighbors and with a 90% train and 10% test split.



Experiment/creative

So, for my creative part I decided to see if I could do some reduction on the telescope data set as it seemed to be a LOT of data and would take a good amount of time to run. My hypothesis was that if there's duplicate data it doesn't contribute much to the algorithm and thus can be removed. My reduction algorithm worked by essentially eliminating similar or duplicative data in the training set and thus reducing the amount of data to check to see if they are neighbors for the test data. I would do this by removing data that was already present within a certain arbitrary range for all attributes. To show an example of this algorithm if I already had information with values 1, 3, 5, 7 and a range of removal of 3 then 1, 2, 6, 9 would not be added but the data sets -2, –5, -10, 10 and 1 10 5 7 would be added. Then for consistency's sake, I ran all of my attempts with this reduced data with 15 nearest neighbors and ran with the telescope test data unaltered. As you can see in the charts below, accuracy gradually goes down the more data that we remove, however we are able to remove 92% of the data and still get an accuracy of 80% which is pretty good. This shows that at least for this data set reduction is definitely worth using as it helps speed up the performance of k nearest neighbors on it. Another thing to consider is that by reducing the data set we might also be able to in turn increase the number of neighbors we check.


0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
Accuracy
0
2000
4000
6000
8000
10000
12000
14000
Data Removed


0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
Accuracy
0
100
200
300
400
500
600
700
800
900
Range