K-Nearest Neighbors and Condensed K-Nearest Neighbors (CS605.449 – Project 3)

By Max Robinson

Abstract

In this experiment the performance of three variations the k-Nearest Neighbor (k-NN) algorithm are analyzed, and two variations are compared to one another. The three variations of k-NN inspected are k-NN regression, k-NN classification, and k-NN condensed. The variations are run on data sets corresponding to the type of problem. Two data sets, Forest Fires and Computer Hardware, are run with k-NN regression. Two other data sets, E coli and Image Segmentation, are run with k-NN classification and condensed k-NN. The results of the experiment showed that the k-NN regression algorithm performed overall poorly on the data set, but proportionally well for the data sets. It also showed that the k-NN classification algorithm out performed the condensed k-NN algorithm, while performing best on the Image Segmentation data set.

Problem

The problem being investigated in this experiment is how well K-Nearest Neighbor (k-NN) performs on four different data sets, using three different algorithm variations. The three algorithm variations are k-NN classification, k-NN regression, and condensed k-NN. The data sets used are the E coli, Image Segmentation, Computer Hardware, and Forest Fire data sets from UC Irvine Machine Learning Repository. Two data sets are for regression, Computer Hardware and Forest Fire, and two are for classification, E coli and Image Segmentation. The other problem being investigated is how well k-NN classification compares to condensed k-NN on the classification data sets

For this experiment, I hypothesize that for the k-NN regression, the MSE will be quite large. The data sets used for k-NN regression in this experiment both have features and labels that have large ranges of values. The Computer Hardware data set has values that have wide ranges and I do not believe it has enough data points in the data set to overcome the sparseness of the data. The Forest Fire data set has a wide range for the labeled values, and in addition has many more zero values than other values. I think this will make it difficult for k-NN regression to behave well on this data set.

For classification, I hypothesize that the error rate will be low for the E coli data set, and that the Image Segmentation data set will perform worse than the E coli data set. The E coli data set has fewer features and appears to be a simpler data set than the Image Segmentation data set which has nineteen features.

In addition, I expect that the condensed k-NN will perform better on each of the classification data sets then the k-NN classification algorithm. The condensed k-NN works to avoid noise in the data and to select the boundary points out of the data to distill the essence of the data set. I think that both of these data sets would benefit from being condensed.

Algorithm Implementation

k-Nearest Neighbor

K-Nearest Neighbors (k-NN) is a lazy, instance based, algorithm that can do either classification or regression. Being instance based and lazy means that k-NN computes a prediction for a query instance directly from the training data that k-NN was supplied, and at the time of querying the model. The actual model for k-NN is the training data that is supplied along with the metric used for calculating distance.

K-NN works by finding k, where k is some number, instances in the model that are the k closest points in the model to the query instance. The distance calculation for the implementation for this experiment is Euclidean distance. Euclidean distance is often used and as described by Aman Kataria as having "a splendid intermingle of ease, efficiency and productivity"[1]. Euclidean distance is defined as $d(q,p) = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}$, for the distance from q to p [1].

After finding the k-nearest points in the model, a function is applied to retrieve the predicted value for the query instance, based on the labels or values of the k-nearest points.

For the implementation for regression in this experiment, which will be referred to as k-NN regression, the function used is a simple average function. This sums the values of the k-nearest data points together and then divides by the number of data points, k. This average is then the predicted value for the query instance.

For the implementation for classification in this experiment, referred to as k-NN classification, a majority vote of the data points is taken. This means, of the k-nearest data points, which ever label is of the k-nearest occurs the most, is the value that is given to the query instance. For tie-breaking, a random rule is selected [1]. This is done by which ever class was found as majority first.

Condensed k-Nearest Neighbor

Condensed k-Nearest Neighbor uses k-NN at query time, but prior to that a smaller model of data points is constructed from the original training. This smaller data set, ds', is then used as the training set for k-NN when a query instance is passed to condensed k-NN.

The condensed data set is constructed as follows.

1. Start with an empty set Z.
2. While Z continues to change, repeat the following:
    a. For each data point in the training set x', find a point x'' in Z that is the closest point in Z to x'.
    b. If the labels of x' and x'' are not the same, add x' to Z

The constructions of Z, which is similar to the CNN described in "Survey of Nearest Neighbor Condensing Techniques", is pointed out to be slow [2]. This was demonstrated anecdotally during the running of these experiments. There are faster ways of creating Z such as RNN or FCNN [2]. These methods were not used in this experiment however due to the added complexity of developing the techniques, in addition to relatively small data sets that are used in this experiment.

In this implementation of condensed k-NN, the k-NN algorithm with a $k = 1$ is used with the data points in Z to calculate the closest point in Z to a given instance when creating the set of condensed points.

This condensed set of data points is calculated every time a new set of training data is used, but is only calculated once. This condensed set is then used for any later query instance. This means that for instance, in cross validation, every time a new training set of data is created from the folds a condensed model is calculated.

Scoring

Mean Squared Error

Mean Squared error is a metric that is used to describe the average squared error for a test set of data when compared to a model. Mean Squared error can be described as the sum of the squared errors for all test data points, divided by the total number of test data points.

$$mse = \frac{\sum_n (predicted\ value_i - actual\ value_i)^2}{n}$$

This error is used to determine how close a predicted regression value is to the actual value when using Euclidean distance as a distance metric.

Error Rate

Error rate is used in this experiment to calculate how many misclassifications occurred when compared to the total number of data points classified. Error rate can be described as

$$errorRate = \frac{\#\ of\ misclassified\ test\ instances}{total\ number\ of\ test\ instances}$$

Experimental Approach

The approach for this experiment was twofold. The first part was done through data preprocessing, and the second part was done through how the experiments were run. The metric used for evaluation are MSE and Error rate for the regression and classification problems respectively.

Data Pre-processing

Each data set used had to be preprocessed to fit the constraints of the algorithms and provide useful data for running the classifications.

For the E coli data set, there were three classes in the data set that had very few examples in the data set. To not skew our results because of a lack of data, these classes were removed from the data set. The classes removed were, omL, imL, and imS. In addition, the data was transformed into csv format with the class label as the last value in the feature vector.

The segmentation data set was originally two different data sets, a training and test set. For this experiment, the two data sets were merged together into one data set so that cross validation could be done on the entire data set. In addition, all class labels were moved to be the last feature in the feature vector.

The Forest Fire data set as manipulated to have all numeric data types. There were two specific features that were non-numeric that were transformed to be numeric, month and day of the week. The month value was replaced with an integer value corresponding to the month number on the calendar instead of a string representation of the month name. For example, “mar” was transformed to “3”. For day of the week, i.e. mon, tue, etc., was given an integer value for each day of the week where Monday was given the value “1”, and Sunday “7”. These transformations allowed all of the data to be handled in the same way and to use the same distance metric.

In the Computer Hardware data set, three feature values were removed. Vendor Name, Model Name, and ERP were all removed. Model Name was removed because the values were highly unique, and as such likely had little effect on prediction. ERP was removed because the field is actually the estimated relative performance that was calculated by the original uses of the data set. These predictions were the result of running their own linear regression algorithms. As such, these values were not the true goal values to estimate and were thus removed.

The Vendor Name was removed for consistency in how distance is calculated for this implementation of k-NN. Distance is calculated using Euclidian distance of numeric values in this implementation. Having non-numeric values would require a different distance calculation. This distance calculation could have repercussions on how the overall distance is calculated and have undesired side effects on the predicted values.

k-NN and Choosing K

There are two types of problems k-Nearest Neighbor can solve, classification and regression. For this experiment two of the data sets used for classification, E coli and Image Segmentation, and two data sets used for regression, Computer Hardware and Forest Fire. When running k-NN for the data sets, the appropriate error function was used. Classification used error rate, while regression used MSE, as described above under the “scoring” section.

For this experiment, cross validation was used to provide robust and averaged results across different distributions of the data. Five-fold cross validation was used for this experiment. This means that from the entire set of the data, five partitions are created. For each run of k-NN the five folds are rotated through, were four of the folds are merged to create training set of 80% of the data, and one fold is used as the test set, 20%. The partition that is used for the test set is rotated so that every partition is used as the test set once.

The data used for each fold in the cross validation are also stratified for classification problems. This means that the distribution of the data for each fold mirrors that of the distribution of the data in the entire training set. If class A makes up 40% of the data in the entire data set, then class A will make up 40% of the data in each fold for cross validation.

This has the effect of running k-NN on five different training and test sets, and producing results for each. These results are then averaged and the standard deviation of the five trials is also calculated. This provides information about how well the algorithm did on average as well has how much it varied between the different test and training sets.

The performance of k-NN relies heavily on what value of k is selected. The procedure for finding the best k value for each data set and each version of k-NN, k-NN or condensed k-NN, was done as follows. For

each data set, k-NN with cross validation was run with a selected value for k. K was allowed to range for k=1 to k=10. This provided results for how well a given k did on a single cross validation run. This was then repeated n number of times, where n varied per data set.

This process provided an average of the error from the cross validations for all k's from one to ten. From this data, charts were created for each combination of data set and k-NN or condensed k-NN that was used to find the best k. The best K was then selected from the charts, where lowest error is best.

The figures below show how the error rate correlates to the number of k's used for each data set with each algorithm, k-NN or condensed k-NN. N in this context is the number of times cross validation was run to get an average of the errors from cross validation.

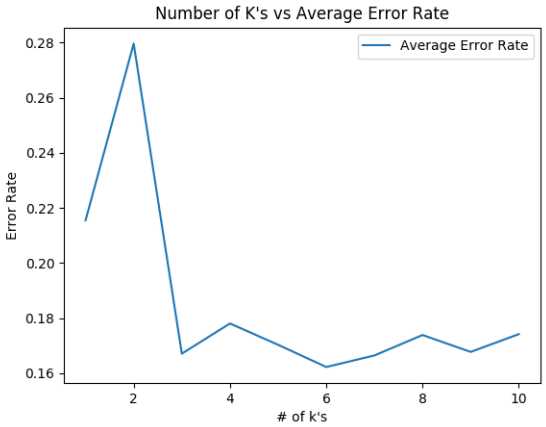

Figure 1 E coli k-NN, k selection, n=20

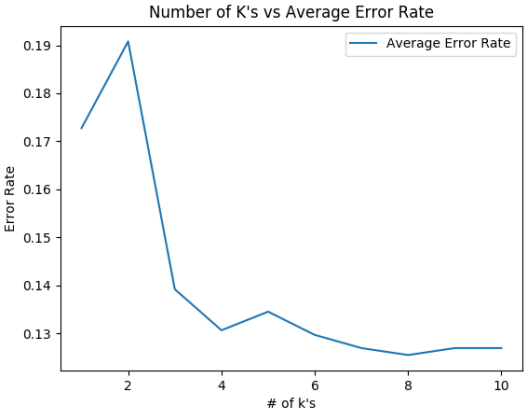

Figure 2: E coli Condensed k-NN, k Selection, n=10

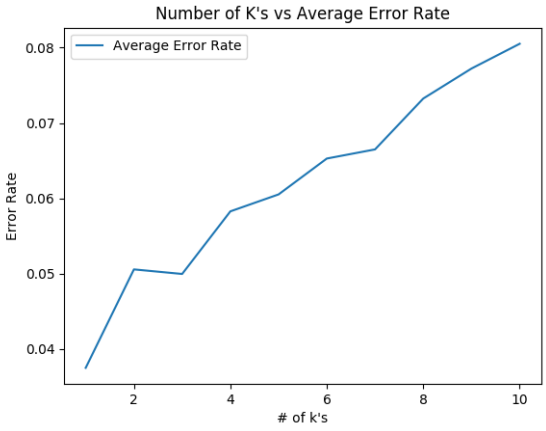

Figure 3 Segmentation Condensed k-NN, k Selection, n=5

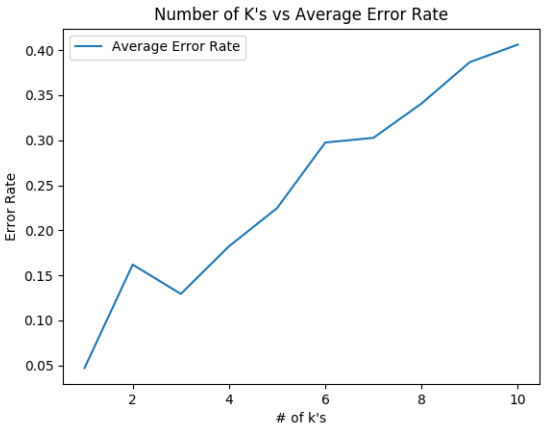

Figure 4 Segmentation Condensed k-NN, k Selection, n=5

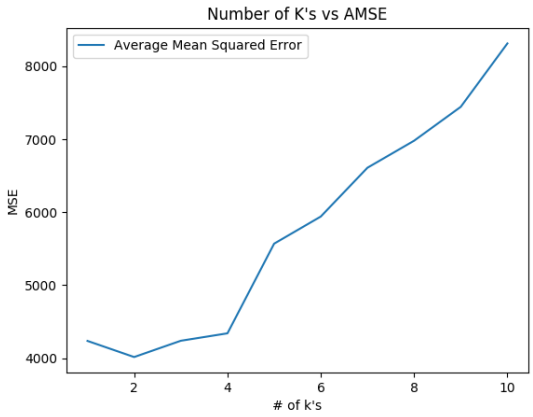

Figure 5 Computer Hardware k-NN, k selection, n=20

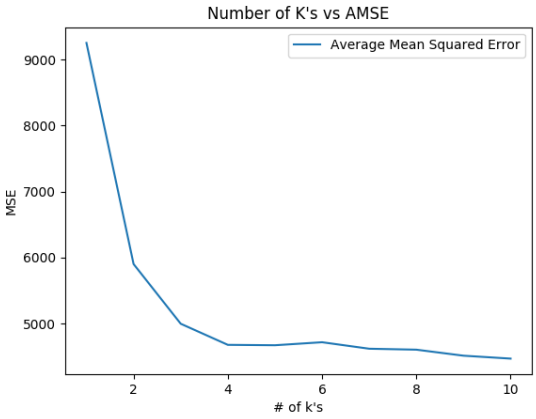

Figure 6 Forest Fire k-NN, k selection, n=20

These figures also show the average error of each k-NN algorithm on each data set for any given k. Table 1 show for which value for k performed best for each data set and algorithm.

Table 1 k value chosen for each data set and algorithm

Dataset	Algorithm	K value chosen
E coli	k-NN classification	K = 6
E coli	Condensed k-NN classification	K = 8
Segmentation	k-NN classification	K = 1
Segmentation	Condensed k-NN classification	K = 1
Computer Hardware	k-NN regression	K = 2
Forest Fire	k-NN regression	K = 10

To calculate the final results, the value for k on each algorithm and dataset was then taken and run directly for the k value with cross validation to generate the output results shown in the results section.

Results

After the running the experiment as described in the above sections, the resulting MSE or error rate for each data set in Table 1 and Table 2 respectively. The error rate for k-NN and condensed k-NN are showed alongside each other to compare the results. These results show the MSE or error rate for k-NN or condensed k-NN with the specified best performing 'k' as shown in the k-NN and Choosing K section.

Table 2 MSE per Data Set with Standard Deviation for Cross Validation

Data Set	Average Mean Squared Error	Standard Deviation of Cross Validation
Forest Fire, k=10	4225.461914368932	4477.235155437042
Computer Hardware, k=2	5062.4390243902435	4597.6086020176635

Table 3 Error Rate to six decimal places per Data Set for k-NN

Data Set	k-NN Error Rate	Standard Deviation of CV
E coli, k = 6	0.129032	0.014426
Image Segmentation, k = 1	0.035931	0.009445

Table 4 Error Rate to six decimal places per Data Set for condensed k-NN

Data Set	Condensed k-NN Error rate	Condensed k-NN Standard Deviation of CV
E coli, k = 8	0.190323	0.041310
Image Segmentation, k = 1	0.049784	0.009285

Behavior

The behavior for the classification algorithms and data set combinations was largely not expected, while the regression combinations behaved close to as expected.

For the regression problems, the results at look to roughly line up with the hypothesis that k-NN regression performed poorly on the two data sets, Forest Fires and Computer Hardware. The Average Mean Squared Error is in the thousands which looks to be quite high. This value is quite large, and it would be expected that it could be lower. This means that the results for both data sets were quite far off from their actual values. However, when compared to the maximum possible error in the data set, the proportions is shown in Table 5.

Table 5 AMSE in proportion to Max MSE

Data set	Average MSE	Max MSE in Data Set	Max MSE/Average MSE
Forest Fires	5062.439	1308736.0	0.00386819
Computer Hardware	4225.46191	1189931.9056	0.00344101

The Max MSE for the data set is calculated by finding the largest distance between two data point's values in the data set, and squaring the difference. This gives us the value of it the actual value had been one of the points and the predicted value had been the point farthest away from that point in the training data.

As a proportion of the maximum possible MSE, k-NN regression performed well. When thinking of MSE in terms of an area of error, k-NN averaged .3% of largest possible area of error. This means that given the data points available in the data set k-NN performed reasonably well in not picking choosing the data points that would produce the largest errors. The total MSE however is still large.

For the classification problems, K-NN classification preformed much better on the Image Segmentation algorithm than expect, while performing much worse on the E coli data set. For Image Segmentation with k-NN classification, there was an error rate of only about 3.5% error, compared to the almost 13% error for E coli data, which was unexpected.

I believe one of the reasons for this is due to both the amount of data as well as the distribution of data in the data sets. The E coli data set had fewer records and the distribution of the classes was not entirely even. The Image Segmentation data had many more examples, and was exactly evenly distributed between all of the possible classes.

When comparing the condensed k-NN to k-NN classification that is not condensed, there is also a difference in what was expected. K-NN classification performed significantly better than condensed k-NN. On the E coli data set, if we compare the error rate of k-NN classification, 0.129032, condensed k-NN, 0.190323, and consider the standard deviation of the k-NN classification, 0.014426, condensed k-NN error rate is more than three standard deviations away from the k-NN classification which is quite far. When considering the condensed k-NN standard deviation though, there is a difference of about 1.5 standard deviations. This is closer, but still shows on average k-NN classification performs better.

For the Image Segmentation data set, the k-NN classification algorithm performed better than the condensed k-NN algorithm on average. When comparing the error rates we see that using either algorithms standard deviation, the error rates are within about 1.5 to 2 standard deviations of each other. This shows that on average, in this case, the k-NN classification algorithm preforms better than the condensed k-NN but it is sometimes possible to have them preform comparably to one another.

Summary

Overall, the algorithms performed differently than what was hypothesized. The regression k-NN, while not performing well in terms of overall MSE, did perform reasonably well when compared to the possible max MSE in the training data. As such this partially matched the hypothesis. On the other hand, k-NN classification performed quite different than expected. K-NN classification performed very well on the Image Segmentation data set, while performing not as well on the E coli data set. When comparing condensed k-NN to k-NN classification for these data sets, k-NN classification performed better across the board.

References

[1] Kataria, Aman; Singh, M.D. A Review of Data Classification Using K-Nearest Neighbour Algorithm. International Journal of Emerging Technology and Advanced Enginee ring: Volume 3, Issue 6. June 2013. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.413.3893&rep=rep1&type=pdf .

[2] Miloud-Aouidate, Amal; Baba-Ali, Ahmed Riadh. Survey of Nearest Neighbor Condensing Techniques. International Journal of Advanced Computer Science and Applications: Vol. 2, No. 11. 2011. https://thesai.org/Downloads/Volume2No11/Paper%2010-%20Survey%20of%20Nearest%20Neighbor%20Condensing%20Techniques.pdf

Data Sources

E coli — https://archive.ics.uci.edu/ml/datasets/Ecoli
Image Segmentation — https://archive.ics.uci.edu/ml/datasets/Image+Segmentation
Computer Hardware — https://archive.ics.uci.edu/ml/datasets/Computer+Hardware
Forest Fires — https://archive.ics.uci.edu/ml/datasets/Forest+Fires