Abstract -

K NN is a non-parametric learning method that may be used for classification or regression. The model may be used in situations where the distribution is known or not known. Through tuning and experimentation the KNN algorithm and variants help reduce the amount of data needed to be analyzed, and their strengths and weaknesses are analyzed.

Problem Statement -

Knowing the distribution of any data set is important information, in cases where this is known parametric learning can be used to mimic the distribution through a set of parameters and forms, an example being Naive Bayes per the last project. These algorithms perform fast and don’t require a lot of data. In this project, the data does not follow a known distribution, therefore, we have to use non-parametric learning to solve this problem, making no assumptions about the data and not constructing a model. This is where the KNN learning algorithm steps in. Our task is to implement KNN algorithm for classification and regression problems, and with the same input file, a secondary task is to reduce the size of the data by implementing Edited nearest neighbor, Condensed Nearest Neighbor, K-Means Clustering, and K-Medoids data reducing Algorithms. What will be of interest will be how the reducers contribute to the performance of K-Nearest Neighbor.

Hypothesis -

KNN is expected to outperform other nearest neighbor methods. KNN uses the most data points form a data set, which effectively minimizes the error rate. In addition, KMeans and K-Medios are expected to perform better than Edited Nearest Neighbor and Condensed nearest neighbor because they will have similar amounts of points, but KMeans and K-Medoids fit the representative points to the known data.

*In the middle of the paper here I think it would be worthwhile and beef up the paper if there was a brief analysis of the algorithms themselves: KNN with Distance Metrics, the reducers; Edited KNN, Condensed KNN, KMeans, and K-Medoids.*

Preprocessing - All examples in the data set are scrambled at random and then assigned to sets for ten-fold CV same as the last project. All categorical variables are converted to integers for convenience. The preprocessor also generates a similarity matrix for each variable to determine distances between categorical variables. All numerical variables are normalized between 0 and 1. The preprocessor should also handle missing variables if need be.

Tuning -

K, the number of NN to compare a query point must be tuned. The value is K=sqrt(N) where N is the number of examples in the data set. For the 10% difference .9\*K and 1.1\*K was tested. Analyzing the results, it should be apparent that for classification sets the learner performs better when using .9\*K than the other possible values for K. For regression sets, there is not a significant change. The other parameter to tune is C, the number of representative points that KMeans and K-Medoids have. We can use MSE and Accuracy for Classification because MSE is a good indication of how far off the predicted distribution is from the actual distribution. Accuracy can show how well the algorithm is classifying examples on an individual basis.

For regression, we will use MSE and Mean Error. MSE will square the difference between real and predicted values. ME will be similar but will not square the difference like MSE. MSE will highlight outliers. ME will show if a learner is over or under through estimating the value contained in the test set. The initial number of clusters will be set to 20%\* the number of points in the data set, then a 10% increase and decrease was tested. It was found that for all the data sets, the difference between each metric value was the same when changing the cluster size, For example, the largest difference obtained in the MSE for Forests Fires was less than 15 when the initial value was over 4000, which is essentially no change. Same with the Mean Error, the largest difference in Forest Fires being .16 when the initial value was around -13, practically no change in the metric. Therefor, changing the number of clusters within 10% of 20%\*Number of Points does not affect the performance of the learner of the regression data sets.

In regard to classification, The initial value for the No. of clusters in a classification set is the value (V) returned by E-NN on the data set. Tested values in 10% increments are 0.9 ∗ V and 1.1 ∗ V. For the glass data set, performance for KMeans did not significantly change. This is likely due to outliers. For K-Medoids, the best performance was 1.1 ∗ V.

Results -

These are the results from my rough runs. The code needs a lot of formatting and commenting work so that will come later, this should be enough to get the report going though.

Glass Accuracy -20%/ -10%/N/+10%+20%

KNN - 78.4/76.3/68.2/70.4/73.4

ENN - 77.7/75.4/68.3/72.4/75.4

CNN - 39.4/40.7/41.2/ 42.7/43.5

KMeans - 7.9/7.9/7.9/7.9/7.9

K Medoids - 67.967.9/67.9/67.9/67.9

Glass MSE -20%/ -10%/N/+10%+20%

KNN - 11.2/10.4/8.7/11.6/11.7

ENN - 10.9/11.4/9.7/12.4/15.4

CNN - 609.3/672.3/1173.3/868.4/954.3

KMeans - 7733.3/ 7733.3/7733.3/7733.3/7733.3

K Medoids - 15.7/14.6/12.4/17.0/18.2

Image Segmentation Accuracy -20%/ -10%/N/+10%+20%

KNN - 83.0/83.3/83.8/81.9/80.2

ENN - 84.3/84.1/83.1/81.5/81.2

CNN - 56.3/54.8/50.0/45.2/42.2

KMeans - 14.3/14.3/14.3/14.3/14.3

K Medoids - 79.8/80.5/82.4/79.5/80.4

Image Segmentation MSE -20%/ -10%/N/+10%+20%

KNN - 2.1/2.1/2.1/2.5/2.5

ENN - 2.0/2.2/2.7/2.9/3.0

CNN - 5.9/6.1/7.8/9.5/9.8

KMeans - 56.4/56.4/56.4/56.4/56.4

K Medoids - 2.9/2.6/2.1/2.9/3.1

Vote Accuracy -20%/ -10%/N/+10%+20%

KNN - 30.9/30.9/30.8/30.9/30.9

ENN - 30.8/31.2/32.3./33.3/33.9

CNN - 29.5/29.9/30.5/30.8/31.3

KMeans - 0.1/0.1/0.1/0.1/0.1

K Medoids - 0.1/0.1/0.1/0.1/0.1

Vote MSE -20%/ -10%/N/+10%+20%

KNN - 238.6/245.3/262.5/299.8/305.2

ENN - 314.9/325.3/335.6/355.9/363.2

CNN - 140.9/148.9/202.3/225.9/236.9

KMeans - 6334.5/6334.5/6334.5/6334.5/6334.5

K Medoids - 6334.5/6334.5/6334.5/6334.5/6334.5

Forest Fires ME -20%/ -10%/N/+10%+20%

KNN - 4239.9/4239.7/4238.9/4238.2/4239.0

KMeans - 4234.7/4235.7/4236.3/4238.2/4239.4

K Medoids - 4229.6/4238.7/4248.27/4238.9/4235.7

Forest Fires MSE -20%/ -10%/N/+10%+20%

KNN - -12.7/-12.7/-12.7/-12.7/-12.7

KMeans - -12.7/-12.7/-12.7/-12.7/-12.7

K Medoids - --12.7/-12.7/-12.6/-12.6/-12.6

Abalone ME -20%/ -10%/N/+10%+20%

KNN - -14.2/-14.3/-14.2/-14.3/-14.2

KMeans - -14.3/-14.3/-14.2/-14.3/-14.3

K Medoids - -14.3/-14.3/-14.2/-14.3/-14.3

Abalone MSE -20%/ -10%/N/+10%+20%

KNN - 199.8/202.4/218.1/231.1/242.3

KMeans - 7056.9/7056.9/7056.9/7056.9/7056.9

K Medoids - 7056.9/7056.9/7056.9/7056.9/7056.9

Machine ME -20%/ -10%/N/+10%+20%

KNN - -0.5/-0.5/-0.5/-0.5/-0.5

KMeans - -0.5/-0.5/-0.5/-0.5/-0.5

K Medoids - -0.5/-0.5/-0.5/-0.5/-0.5

Machine MSE -20%/ -10%/N/+10%+20%

KNN - 2.1/2.1/2.1/2.1/2.1

KMeans - 2.1/2.1/2.1/2.1/2.1

K Medoids - 2.1/2.1/2.1/2.1/2.1