Implementation of K-Nearest Neighbors and its variants on classification and regression datasets

Mohammed Momin

1. Introduction

The problem of classification and regression is fundamental in machine learning. For each dataset however, a different algorithm may suit it better than one that is the best for a prior dataset. Even within classification algorithms themselves, there can be different variants that may change the results. One of those algorithms is integral to this project: The K-Nearest Neighbors (KNN). The algorithm is a non-parametric classification method that is also used for regression. In both cases, the input is the k closest training examples in the set.

While the KNN algorithm is great by itself, data reduction is one of the most important problems when dealing with large classification datasets. That is where Condensed Nearest Neighbor (CNN) is helpful. CNN decreases the size of the training set by adding all of the misclassified elements in an empty set and classifying on that instead. The Edited Nearest Neighbor (ENN) also decreases the size of the training set but instead it uses all of the elements that are classified correctly. In this project, I tested to see which of these 3 algorithms worked best on various datasets.

Hypothesis:

Out of the 3 different versions of the classification algorithm, I think the normal KNN algorithm would perform the best despite having a higher run time since it uses all of the dataset. If we had to use data reduction, I think the ENN algorithm would perform better than the CNN because I think there would be more elements that are classified correctly than incorrectly.

2. Experimental Approach

In the beginning of each algorithm, 5 equal folds are made of the data. Essentially, the algorithm runs 5 times where each of the folds has a turn being the test set and the rest of the folds being the training set in each run.

The algorithm for KNN classification is for every test point:

1. Compute the Euclidian distance between the test point and every training point
2. Select the k closest and their classification
3. Output the class that is most frequent amount the k closest neighbors

Certain assumptions made by my algorithm is that the actual class column is always the last one of a classification dataset and it is numerical. The accuracy of the algorithm is calculated by the percentage of correct output classes predicted. An accuracy metric was given for each of the five folds or splits that are made and then an average of the metrics is taken for the accuracy of the algorithm as a whole. This was used to tune k has different k values were iterated through and the k value giving the best accuracy score is deemed to be the best.

For the CNN and ENN, the training set is modified before implementing the KNN algorithm. In CNN, the algorithm classifies a training set and only takes the training points that were classified correctly and uses that as a training set. In ENN, the algorithm classifies a training set and only takes the training points that were classified incorrectly and uses that as a training set. The K for this algorithm is tuned the same was as KNN in an iterative approach to get the highest accuracy.

The K Nearest Neighbor algorithm I also used for regression datasets. To do that, KNN finds the k closest training points to a test point being regressed and uses the Guassian kernel to

take a weighted average of those neighbors by the distance. The kernel density estimate is given as follows:

$$p(\mathbf{x}) = \frac{1}{N}\sum_{n=1}^{N} k(\mathbf{x}, \mathbf{x}_n) \qquad k(\mathbf{x}, \mathbf{x}_n) = \frac{1}{(2\pi h^2)^{D/2}} \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}_n\|^2}{2h^2}\right)$$

Here, the h is manually tuned in the algorithm to find the optimum performance.

Datasets and preprocessing:

1. Glass [Classification]

   The study of classification of types of glass was motivated by criminological investigation. For this dataset, the "Id" column was removed and the missing values were replaced with NaN.

2. Image Segmentation [Classification]

   The instances were drawn randomly from a database of 7 outdoor images. The images were hand segmented to create a classification for every pixel. For this dataset, the class classification column had to be moved to the end of the array instead of it being the first column. The classification had to be converted from string to numerical value.

3. Vote [Classification]

   This data set includes votes for each of the U.S. House of Representatives Congressmen on the 16 key votes identified by the Congressional Quarterly Almanac. For this dataset, I had to convert all the y's to 1s and n's to 0s. All of the ? were replaced with NaN values. The class name ("Democrat" or "Republican") was also made numerical (0 or 1).

4. Abalone [Regression]

Predicting the age of abalone from physical measurements. For this dataset, the only preprocessing that was done was converting the gender column to a binary numerical column

5. Computer Hardware [Regression]

   The estimated relative performance values were estimated by the authors using a linear regression method. To preprocess this data, I converted the Vendor Name and Model Name to a numerical variable based on a unique key.

6. Forest Fires [Regression]

   The aim is to predict the burned area of forest fires, in the northeast region of Portugal, by using meteorological and other data. To preprocess this data, I converted the month and day to a numerical variable instead of string.

4. Results for best K value for a given dataset

a. Glass [Classification]

```
------------------------------------------------------------
Algorithm name: K-Nearest Neighbor
Accuracy Statistics for All 5 Experiments:
[0.75555556 0.68181818 0.81395349 0.69047619 0.775     ]
Problem Type : Classification

Value for k : 1

Classification Accuracy : 74.3360683244404%

------------------------------------------------------------
Algorithm name: Condensed K-Nearest Neighbor
Accuracy Statistics for All 5 Experiments:
[0.68888889 0.68181818 0.79069767 0.71428571 0.575     ]
Problem Type : Classification

Value for k : 1

Classification Accuracy : 69.01380918822278%
```

```
------------------------------------------------------------
Algorithm name: Edited K-Nearest Neighbor
Accuracy Statistics for All 5 Experiments:
[0.68888889 0.63636364 0.69767442 0.64285714 0.8       ]
Problem Type : Classification

Value for k : 1

Classification Accuracy : 69.3156817342864%
```

b. Image Segmentation [Classification]

```
------------------------------------------------------------
Algorithm name: K-Nearest Neighbor
Accuracy Statistics for All 5 Experiments:
[0.80952381 0.95238095 0.95238095 0.9047619  0.88095238]
Problem Type : Classification

Value for k : 1

Classification Accuracy : 90.0%


------------------------------------------------------------
Algorithm name: Condensed K-Nearest Neighbor
Accuracy Statistics for All 5 Experiments:
[0.88095238 0.88095238 0.88095238 0.83333333 0.88095238]
Problem Type : Classification

Value for k : 1

Classification Accuracy : 87.14285714285715%


------------------------------------------------------------
Algorithm name: Edited K-Nearest Neighbor
Accuracy Statistics for All 5 Experiments:
[0.88095238 0.88095238 0.9047619  0.88095238 0.85714286]
Problem Type : Classification

Value for k : 1

Classification Accuracy : 88.09523809523809%
```

c. Vote [Classification]

```
------------------------------------------------------------
Algorithm name: K-Nearest Neighbor
Accuracy Statistics for All 5 Experiments:
[0.76136364 0.82954545 0.81609195 0.86046512 0.86046512]
Problem Type : Classification

Value for k : 9

Classification Accuracy : 82.558625549804 38%


------------------------------------------------------------
Algorithm name: Condensed K-Nearest Neighbor
Accuracy Statistics for All 5 Experiments:
[0.85227273 0.75       0.83908046 0.88372093 0.8255814 ]
Problem Type : Classification

Value for k : 12

Classification Accuracy : 83.01311025248474%


------------------------------------------------------------
Algorithm name: Edited K-Nearest Neighbor
Accuracy Statistics for All 5 Experiments:
[0.81818182 0.82954545 0.82758621 0.79069767 0.8255814 ]
Problem Type : Classification

Value for k : 8

Classification Accuracy : 81.83185098782533%
```

d. Abalone [Regression]

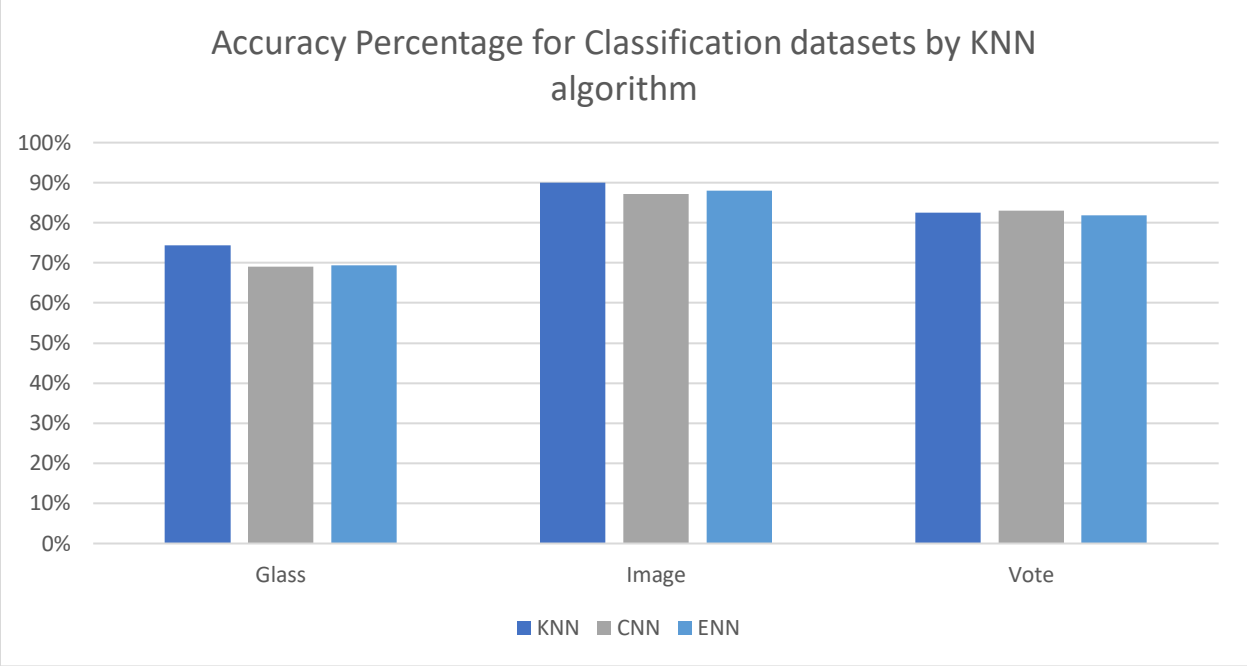
```
------------------------------------------------------------
Algorithm name: K-Nearest Neighbor
Accuracy Statistics for All 5 Experiments:
[9.53793593 8.28553048 8.35182829 8.00396521 8.12933175]
Problem Type : Regression

Value for k : 1

Mean Squared Error : 8.461718331981777
```

e. Computer Hardware [Regression]

```
------------------------------------------------------------
Algorithm name: K-Nearest Neighbor
Accuracy Statistics for All 5 Experiments:
[59510.58048071  7667.34024507  4031.77098825  1243.11966672
  1238.70247854]
Problem Type : Regression

Value for k : 30

Mean Squared Error : 1473830.2771857672
```

f. Forest Fires [Regression]

```
------------------------------------------------------------
Algorithm name: K-Nearest Neighbor
Accuracy Statistics for All 5 Experiments:
[59510.54607882  7667.34042543  4031.77497547  1243.49999999
  1238.73233732]
Problem Type : Regression

Value for k : 1

Mean Squared Error : 1473837.876340601
```



5. Discussion

The generalized KNN algorithm generally outperformed the CNN and ENN algorithm. Which makes sense since the data would be trained on a largest dataset than a smaller one. However, it was much more time consuming to run the regular KNN opposed to the CNN or ENN and while the CNN and ENN algorithms performed worse, they were not that much worse as shown by the bar chart above.

Both the CNN and ENN algorithms had relatively the same accuracy percentage throughout the board but it was interesting to see that some datasets were classified much better in general

than others. For example, the image classification dataset had an accuracy of 85%+ whereas classifying the glass data was less than 75% accurate. I wonder why this may be the case. Is it the structure of the data itself or the features that were selected in each experiment?

The mean squared error for 2 of the datasets (forest fires and computer hardware) being regressed was also very high compared to the Abalone dataset. This could be for a number of reasons including the dataset having a high variance measure. This was also manually tuned in the calculation of the Gaussian Kernel.

6. Conclusion

My hypothesis of the normal K Nearest Neighbors algorithm performing the best was proven to be correct for the datasets given. The data did not show a preference between the Edited Nearest Neighbor and Condensed Nearest Neighbor as their accuracy was relatively the same. The classification algorithm worked the best on the image segmentation dataset and the regression algorithm worked the best on the Abalone dataset based off of the best accuracy or mean squared error provided. The importance of data reduction was seen due to the difference in runtime of the KNN algorithm vs the ENN or CNN algorithm. While the Edited Nearest Neighbor and Condensed Nearest Neighbor algorithm performed slightly worse than the normal K Nearest Neighbor algorithm, both, ENN and CNN, had a much better runtime than KNN.