Project 2: *k* Nearest Neighbors

Sarah Wilson SWI1S117@JHU.EDU
303-921-7225
Engineering Professionals Computer Science
Johns Hopkins University
Baltimore, MD 21218, USA

1. Introduction

Regression and classification are both common tasks in the realm of Machine Learning. Regression and classification are both supervised learning problems. Supervised learning is where the system is given an input and output and then asked to learn or predict the mapping of input to output. The algorithm explored in this paper is K-Nearest Neighbor (KNN). KNN is an example of a nonparametric algorithm, nonparametric algorithms do not make any strong assumptions about the form of the mapping function from input data to output predictions. The advantage offered by this approach is that not a lot of prior knowledge on the data or its features is required to build a predictor. KNN is nonparametric as it makes predictions for new data based upon training data by looking at the k closest neighbors to the new data. The primary philosophy behind the kNN algorithm is ””(INSERT REF)

INSERT REFERENCES ON APPLICATIONS

The problem statement presented in this paper is to understand and implement a kNN classifier and regressor on 6 different and unique data sets. The experimentation will first be tuned using a validation subset of the overall data set under experimentation, to determine the most optimal value of k to use. Then the optimal value of k will be run through the full k-fold cross validation process. The experimentation will examine: kNN, edited kNN and condensed kNN The results presented will be the classification error and the regression mean squared error on each of the 6 unique data sets and across the 3 variations of the kNN algorithm.

The hypothesis of this report is that INSERT

Section 1 has provided the introduction, problem statement and hypothesis in regards to the kNN algorithm. Section 2 will provide an in-depth explanation of the kNN algorithm, how the algorithm will be tuned and specifics on each of the 6 data sets used. Section 3 will present the results obtained by variations of kNN, edited kNN and condensed kNN and the values that were chosen as part of the tuning process. Section 4 will discuss the results that were obtained and compare them to the hypothesis that was outlined in the introduction. This report will conclude in Section 5 with a discussion of lessons learned and areas of possible future work.

2. Algorithms and Experimental Methods

The experimental method used in this report is k-Fold Cross Validation. k-Fold Cross Validation is used when the data sets that an algorithm is being experimented on is small, the goal of k-Fold Cross validation is to maximize the amount of data that is used for training of the algorithm. The expriment will use 5-Fold Cross Validation ($k = 5$).
For the experiment in this report a Validation / Tuning set is first used to determine the optimal value of k neighbors in the kNN.

General *k*NN

The approach is outlined in the steps below for the General kNN algorithm implementation:

```
Step 1. Load in Entire data set
Step 2. Randomly shuffle the Entire data set
Step 3. Start Validation / Tune Process
    Step 3.1. Remove 20% of data points from Entire data set, assign to new Validation / Tune data set.
    Step 3.2. Assign a list of $k$ neighbors values to compare for optimization.
    Step 3.3. Break Validation / Tune data set into 5 folds.
    Step 3.4. For each of the $k$ neighbors in the list for optimization:
    Step 3.5.     For a fold to maximum number of folds (5):
    Step 3.6.     Assign 1 fold to be the Test Set
    Step 3.7.     Assign and concatenate the other 4 folds to be the Train Set
    Step 3.8.         For each Data Point in the Test Set:
    Step 3.9.             Assign the current Data Point as the Query Point
    Step 3.10.                For each Data Point in the Training Set:
    Step 3.11.                    Calculate the distance between the Query Point and the current Data Point in the Training set
    Step 3.12.                    Store distance in an overall list for Query Point sorted from smallest to largest.
    Step 3.13.                Get the Predictor for each of the $k$ closest neighbors as determined by the distance list for the Query Point.
    Step 3.14.                If Classification:
    Step 3.15.                    Return the most common predictor
    Step 3.16.                If Regression:
Step 4. Run $K$NN with Optimal $K$ Neighbors Value
    Step 4.1. Using remaining 80% data points from Entire data set, assign to new Train / Test data set.
```

Step 4.2. Repeat steps XXX - XXX.
Replacing the list list of k Neighbors values with the single optimal K Neighbors value.

Edited *k*NN

The approach is outlined in the steps below for the Edited kNN algorithm implementation:

Condensed *k*NN

The approach is outlined in the steps below for the Condensed kNN algorithm implementation:

Data Sets

The following data sets were used during the classification and regression tasks for this project.

Breast Cancer

Description:
Task: Classification
Predictor: Diagnosis (Malignant or Benign)
Link:
`https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Original%29`

Car Evaluation

Description:
Task: Classification
Predictor: Car Evaluation (Unacceptable, Acceptable, Good, Very Good)
Link:
`https://archive.ics.uci.edu/ml/datasets/Car+Evaluation`

Congressional Vote

Description: 1984 United Stated Congressional Voting Records
Task: Classification
Predictor: Party (Republican / Democrat)
Link:
`https://archive.ics.uci.edu/ml/datasets/Congressional+Voting+Records`

Albalone

Description: Physical measurements of Albalone
Task: Regression
Predictor: Rings (int)
Link:
`https://archive.ics.uci.edu/ml/datasets/Abalone`

Computer Hardware

Description: Relative CPU performance data.
Task: Regression
Predictor: PRP
Link:
`https://archive.ics.uci.edu/ml/datasets/Computer+Hardware`

Forest Fires

Description: Forest Fire burn area data
Task: Regression
Predictor: Area (float)
Link:
`https://archive.ics.uci.edu/ml/datasets/Forest+Fires`

3. Results

The following results were obtained from the Classification task data sets. Tables 1-3 display the results from the Breast Cancer, Car Evaluation and Congressional Vote data sets. These tables show the result from the train set and the test set during each fold of the k-fold validation process. The tables also display the accuracy and error obtained during each fold, and the total accuracy and error as averaged across each of the 5 k-folds. An accuracy(scale of 0-1) of 1 indicates that the train set had the most frequently occurring classifier equal to the most frequently occurring classifier in the test set. An error(scale of 0-1) of 0 indicates that the test set and train set had the same most frequently occurring classifier. Note in the Breast Cancer data set a value of 2 was mapped to the classifier of benign.

Table 1: Breast Cancer: Naive Majority Predictor Results

Table 2: Car Evaluation: Naive Majority Predictor Results

Table 3: Congressional Vote: Naive Majority Predictor Results

The following results were obtained from the Regression task data sets. Tables 4-6 display the results from the Albalone, Computer Hardware and Forest Fire data sets. These tables show the result from the train set and the test set during each fold of the k-fold validation process. The tables also display the error obtained during each fold, and the total error as averaged across each of the 5 k-folds The error is the absolute error and was calculated as: $|TestSetAverage - TrainSetAverage|$. This is meant as a measure of how far off the train set average was from the test set average.

Table 4: Albalone: Naive Mean Predictor Results

Table 5: Computer Hardware: Naive Mean Predictor Results

Table 6: Forest Fire: Naive Mean Predictor Results

4. Discussion

INSERT

5. Conclusion

INSERT