JHU AAP EN.605.649.83.FA21 Project 2 Report:

K nearest neighbor classification: normal, edited, condensed

Mauro Chavez Mauro.antoine.chavez@gmail.com | MCHAVEZ8@JH.EDU
AAP M.S. Bioinformatics Student *(858)354-7465*
San Diego, CA

Abstract

This document summarizes the second project deliverable for Introduction to Machine Learning taken through Johns Hopkins AAP in Fall of 2021. This project involved implementing K-nearest neighbor (K-nn) and applying it to both regression and classification tasks. Edited K-nn and condensed K-nn were implemented as well in order to reduce the training data set size and resulting computational load of classification. K-nn required testing in order to identify optimal K's for each data set. Additionally, regression tasks required testing various bandwidth parameters to find one that maximized prediction ability as measured by mean squared error. 6 data sets from the UCI Machine Learning Repository were used in this investigation. Testing showed that smaller K's resulted in better predictive power which may be a result of the test data sets being curated and limited in size.

Video demonstration of code can be found here: https://youtu.be/G68l3St5wEQ

1 Introduction

This assignment had students implement a nonparametric algorithm and perform both classification and regression on data sets from project 1. For classification tasks, the breast cancer, car evaluation, and congressional votes data sets were studied. For regression tasks, the abalone, computer hardware, and forest fire data sets were studied. K nearest neighbor (K-nn)-based approaches were the approach taken in this assignment where edited and condensed variations were implemented as well. In the case of classification, predictions were made using a plurality class vote of a query point's K neighbors. For regression, a Gaussian kernel was used amongst a query points K neighbors to assign a class value. Algorithms and supporting functions were implemented following test-driven development. Learning was assessed using 5-fold cross validation.

1.1 Problem Statement

The high-level goals of this assignment were to implement a K nearest neighbor classifier that could handle both classification and regression tasks in addition to its optimized variations that utilize an edited or condensed training set. Distance in classification tasks was calculated using Euclidean distance. Regression tasks applied a gaussian kernel to the Euclidean distance with the following equation where D(X, Y) returns the Euclidean distance between point X and Y.

$$K(X_q, X^t) = \exp\left[\frac{1}{-2\sigma} D(X_q, X^t)\right]$$

Sigma represents a bandwidth parameter and was a value that had to be tweaked using a hyperparameter validation set made up of 20% of the data partitioned out prior to 5-fold cross

validation of K nearest neighbors. 5 different options for sigma was tested for each regression data set. To assign a value to a point when doing regression, the following equation was applied to accomplish a radial bases function adjusted weighted average.

$$Prediction(X_q, Neighbors) = \frac{\sum_{i=1}^{n} K(X_q, X_i^t) * Label(X_i^t)}{\sum_{i=1}^{n} K(X_q, X_i^t)}$$

In applying K nearest neighbor to each data set, part of the protocol included tuning runtime parameters like K, allowed error, sigma by testing multiple options and evaluating the resulting efficiency of the model. For classification tasks, classification accuracy was used to quantify algorithm effectiveness by calculating the proportion of data points where the predicted label matched the actual label. For regression tasks, classification accuracy was calculated as well except a 10% of the average class value error was allowed when determining if a point was classified correctly. More importantly, mean squared error was calculated for each test point and an average mean squared error was used to quantify effectiveness.

Once effective parameters were studied for each data set, edited and condensed nearest neighbor approaches were applied to see how decreasing training set size effected K nearest neighbor's predictive power.

1.2 Hypothesis

For these assignments, we use curated data sets made available by the UCI Machine Learning Repository (https://archive.ics.uci.edu/ml). Given this context, it may be more likely that neighboring datapoints are in fact true correct neighbors. In other words, errors in data collection or outlier datapoints will be less common in the training data as opposed to data sets sourced more organically. As a result, when looking at a set of neighbors, it is more likely that the neighbors actually belong together and noise from dissimilar datapoints will be minimized. Therefore, smaller K's will result in better predictive power than larger ones. A larger K would open predictions up to influence from data points outside of the assumed tight grouping of datapoints within a common class. Furthermore, our data sets are fairly small. This means that larger K's will be even more detrimental as a larger K may select points from neighboring different classes and hurt predictions.

2 Data Pre-Processing

As this learning approach relied on being able to calculate the Euclidean distance between points, all attributes of the data had to be numeric. For classification tasks, the class label could be either numeric or string, but for regression tasks, the class label had to be numeric. This required some extra tuning and normalization of the data prior to learning.

Defining the folds used in K fold cross validation was challenging for regression-based activities as this study attempted to define folds such that the distribution of labels in each remained comparable to the distribution of labels in the initial data set. This approach was much more feasible for discrete labeled data sets as data points would overlap in their labels. However, for continuous labeled data sets, this almost never happened. In order to create folds with comparable label distribution to the initial data set, continuous labels were placed into 10 bins of equal width. After labels had been binned, then the folds of the data were defined for both the K fold cross validation and hyperparameter tuning sets. Once the indices of the validation and hyperparameter sets were defined, the binned label column was dropped from the data frame and learning continued.

2.1 Abalone Data

Abalone data was used for regression where attributes were used to predict age. Age appeared in the data set as the rings column as ring counting is used as a proxy for age. The "sex" attribute of this data set was broken out as nominal data resulting in sex_m and a sex_f columns.

2.2 Breast Cancer Data

Breast cancer data was used for classification where attributes were used to predict cancer class. Cancer class for this sample was either 2 for benign or 4 for malignant. There existed some "?" values in this data set. These were replaced with null values and imputed using the column mean. The sample column was dropped as data points were instead identified by indices in the data frame.

2.3 Car Data

Car data was used for classification in order to predict acceptability. Similar to the first assignment, the buying, maint, safety, and acceptability attributes were normalized into ranks where 0 was the best/most desirable. Normalizing the acceptability column here was unnecessary for classification purposes as labels would be assigned using a plurality vote of neighbors. However, the adjustment was still made in case data wanted to be plotted in such a way to view how acceptable the entire set of cares is in ranked fashion. With respect to the doors, persons, and lug_boot attributes, these were broken out into nominal data columns.

2.4 Forest Fire Data

Forest fire data used for regression in order to predict burnt area. To start, the month and day attributes were broken down into nominal data columns. It was recommended in the data description file for this set that the area be log transformed as values are heavily skewed towards 0.00. However, when $\log_{10}$ transforming this column, all 0.00 values would be replaced with -inf in the data frame which was not useful for regression. To correct for this, a noise factor of

0.0000001 was added to each data point's area attribute. After this adjustment, log transformation gave non infinite values to each data point.

2.5 House Votes Data

The House votes data set was used for classification where voting history was used to predict whether there would be a yes vote on the crime bill. Every attribute was broken down into nominal data columns.

2.6 Computer Hardware (aka Machine) Data

The computer hardware data set was used for regression where the estimated relative performance was used as the target of prediction. The vendor and model_name attributes were dropped as they did more to serve as data point naming than actual device performance quantification.

3 Experimental Approach

3.1 Tuning K for classification

Each data set used for classification will be tested using K nearest neighbor for 5 values of K. For these tests, no changes will be made to the training set via editing or condensing. Each K will be tested using 5 fold cross validation where the average classification accuracy score will be reported.

3.2 Determining accuracy tradeoff of adjusted training sets

Once the optimal K for each classification data set has been identified using classification accuracy from unedited K nearest neighbors amongst the 5 tested values as described in section 3.1, those will be used in both edited and condensed nearest neighbor. Both edited and condensed nearest neighbor algorithms will be applied to each data set using their optimal K under 5-fold cross validation. The classification accuracy will be averaged across each fold for condensed and edited nearest neighbor. From there, full K-nn, edited K-nn, and condensed K-nn will be compared to determine the cost in terms of classification accuracy to decreasing the training set.

3.3 Tuning bandwidth parameter (sigma) for regression

For regression tasks, the gaussian kernel requires a bandwidth parameter (sigma) to define how the neighboring values are weighted when predicting a value for a query point. For the regression data sets, 5 sigma values proportional to the data set size and label values will be selected and tested on 20% of the data set aside prior to 5-fold cross validation partitioning of the remaining 80%. The mean squared error for each sigma will be computed. The sigma with smallest mean squared error will be selected for further regression tasks.

4 Experimental Results

4.1 Results of tuning K for classification

	3	5	15	25	50	100
Breast Cancer (~700 dp)	N/A	**0.9629173500**	0.9615293054310	0.9558351373612	0.8184338851107	0.6552241003549
Car Evaluation (~1700 dp)	N/A	0.7514450867	**0.7890173410404**	0.7341040462427	0.6791907514450	0.638728323699
Congressional (~450 dp)	**0.88507205171**	0.8642742582	0.848049257612	0.8480225267915	0.84580265364146	N/A

Table 1: Results of running classification data sets with different K's reporting the average classification accuracy from 5-fold cross validation.

4.2 Results of determining accuracy tradeoff of adjusted training sets

	K-nearest neighbor	Edited K-nn	Condensed K-nn
Breast Cancer (classification, k=5)	0.962917350010569 (runtime 6min)	0.9629376134350878 (runtime 60 min 24 s)	0.7907774449133702 (runtime 1min 35s)

Table 2: Results of running K-nearest neighbor, edited K-nearest neighbor, and condensed K-nearest neighbor reporting the average classification accuracy from 5-fold cross validation where training set is recalculated for each fold.

4.3 Results of tuning bandwidth parameter (sigma) for regression…

	0.5	1.5	5	10	20
Abalone	**11.010215710570433**	11.157833738632146	11.210407178098784	11.221802321388921	11.227520329330742

Table 3: Abalone data set – resulting mean squared error while tuning sigma for regression with k = 20 using K-nearest neighbor.

	20	**100**	300	750	1000
Computer Hardware	141295.66690892627	**124309.68404586214**	125088.2887875917	126159.88800088143	126921.29862413228

Table 4: Computer hardware data set – resulting mean squared error while tuning sigma for regression with k = 20 using K-nearest neighbor.

	1**0000**	100000	1000000	10000000	100000000
Forest Fires	**15.787991742833684**	15.791818124012222	15.792207423370217	15.792246420369066	15.792250320740031

Table 5: Computer hardware data set – resulting mean squared error while tuning sigma for regression with k = 20 using K-nearest neighbor.

5 Discussion

With respect to testing Ks for classification tasks, table 1 shows that smaller Ks were favored as opposed to larger ones. The congressional dataset had best performance with the smallest K and interestingly it is the smallest data set. A value of K=3 was tested for this data once the results of K=15, K=25, and K=30 revealed the floor of classification accuracy. The breast cancer data set being almost twice the size of the congressional one had best performance for the smaller Ks (5/15/25). It also had by far the biggest drop off as K increased. This is likely a result of both the data set size being smaller than the car data set and the class labels only occupying two values, 2 and 4. The car evaluation had marginally better predictive power at k=15 than k=5 before classification accuracy began to decrease at K>=25.

With respect to quantifying accuracy tradeoff from applying different versions of K-nn the results are disappointing. Edited K-nn was ran such that for each fold, the training set was re-edited. This may have been a mistake, but it seems correct in terms of methodology. However, the runtime on the breast cancer data set broke an hour in order to perform 5-fold cross validation. This meant losing time that I would have liked to spend doing the same algorithm comparison across other data sets. Unfortunately, there was a negligible increased in classification accuracy (+0.00002) so increased runtime was not rewarded with identifying a far superior algorithmic approach. When

comparing the runtime of edited K-nn to K-nn or condensed K-nn, edited K-nn may appear to lose its utility. However, it's runtime only suffered as much as it did because classification accuracy had to be validated across 5 folds. In an application where the training set can be edited once and saved for future classification activities, edited K-nn is likely a better choice than K-nn as classification will be quicker on account of the smaller training set. This is before considering the potential for accuracy to be increased which is supposed by table 2. Condensed K-nn shines as by far the quickest algorithm when both shrinking a test set on the fly and using it to classify. In a situation where the training set can be edited prior to predicting, edited K-nn outperforms condensed K-nn. However, if training set must be edited and used for classification rapidly, condensed K-nn provides increases in runtime with reasonable accuracy tradeoffs.

When it came to tuning sigma for regression, I had the most success with the abalone and computer hardware data sets. The abalone data set had a narrower range of class values than the computer hardware data set and as a result a much tighter sigma made more sense. As shown by table 3, from sigma=0.5 to sigma=5 there is an order of magnitude change and yet only a 0.19 increase in mean squared error. For the computer hardware data set, a wider range of sigma's were tested. 100 showed to be the best from testing but there may exist a more optimal sigma between 100 and 300 as shown by table 4. The forest fire data set was by far the hardest to study. The labels had undergone the correction described earlier having a noise factor added prior to being log transformed. The sigma's tested for in forest fire regression grow in orders of magnitude with relatively negligible changes in mean squared error.

6 Conclusion

The component of my hypothesis that smaller K's work better for the curated data sets used in this analysis seems to hold as shown by the results in table 1. However, I ran out of time to do the same analysis with the regression data sets once optimal sigma's were identified. A comprehensive review of both classification and regression tasks may show that smaller K's may not always be the better option. It's important to note that with respect to regression, the bandwidth parameter will add a ceiling to K where past a certain point, a larger K will not change predictive power. This occurs because past a certain distance, the weighting of a value to a prediction will result in it having little effect on the regression calculation. Edited K-nn seems powerful when a training set can be trimmed and saved prior to classification as running through 5-fold cross validation and recomputing the training set for each iteration is very costly in terms of computation time. Condensed K-nn provides gains in decreased run time with reasonable tradeoff in classification accuracy. However, I only had time to test this on the breast cancer data set and the other data sets may tell a different story. Testing of sigma parameters showed that these can be tweaked and optimized to get better results except for the case of the forest fire data. My normalization to the class values may have removed any benefits that come from a tuned sigma value.

When running various analysis for this project, even though data sets were small, runtimes were large given the amount of time allotted for completing this assignment. There are three factors to this setback. The first is that my implementation utilized pandas Dataframes. These data structures are good for performing tasks in aggregate. Dataframes excel when doing selections or merges. However, they suffer when they are iterated through row by row. The implementation of algorithms in this project really suffered when expanding to the test data sets. For example, when identifying the K nearest neighbors of a point, all other points in the data frame had to be checked. Classifying n points on a training set of m points then has complexity n^m. This is only further complicated when considering something like edited nearest neighbor where each data point in the training set must first be classified using the rest of the training set before any classification on a test set can begin.

As a result, time was lost to computation that would otherwise have been spent doing data investigations. The third factor being that I could have also started this assignment earlier. Given more time, I would have liked to do the same k-nn/edited k-nn/condensed k-nn comparison for each of the regression as well once optimal sigmas were identified. While I could have given myself more time for this assignment, I still invested many hours into it. When curating my report, I found it hard to manage the combinatorial complexity of testing each data set with various runtime parameters and various flavors of K-nearest neighbor learning algorithms.