Project 2: K-Nearest Neighbor

Dongjun Cho DCHO13@JH.EDU
Programming Project 2
605.649
19 February 2022

Abstract

This project is intended to implement the K-Nearest Neighbor Algorithm for classification and regression. This algorithm was applied to discrete and continuous-valued data using six data sets obtained from the UCI Machine Learning Repository.

1. Introduction

K-nearest neighbor is one of the supervised learning algorithms to solve classification and regression problems. There is a proverb that says "Birds of a feather flock together." It means those who have similar characteristics or attributes, tend to form groups. KNN algorithm works the same way as the proverb. This algorithm works based on the basic assumption that similar things exist close to each other.

The purpose of this paper is to research and implement the non-parametric algorithm to perform classification and regression. In this paper, we will implement standard KNN, edited KNN, and condensed KNN for classifier and regressor to check how these algorithms impact accuracy. The goal of the edited KNN and condensed KNN is reducing the size of the training set and decreasing the noise in the training set. Both edited KNN and condensed KNN will outperform noisy data. In General Cases, Standard KNn will outperform other KNN algorithms.
Hypothesis 1: We hypothesize that standard KNN will outperform edited and condensed KNN. To test this hypothesis, we compared the classification accuracy score for classification and mean squared error for regression tasks on several datasets from the UCI repository.
Hypothesis 2: We hypothesize that both condensed and edited KNN will reduce the execution time for standard KNN because it reduces the size of the dataset.

In Section 2, we describe how each algorithms works, in section 3, we describe the experimental approach to the algorithm, in section 4, we present the result of our experiment. We discussed the behavior of the algorithm in section 5, and then we conclude the discussion with the result and possible future works in section 6.

2. Description of Algorithms

2.1 K Nearest Neighbors

KNN is a non-parametric algorithm, so it has a small number of parameters that can be inferred from the data. It is also a Lazy model which means it doesn't do anything about fitting the data. It finds k neighbors with similar characteristics and has classes of those neighbors as classification values, or calculate the average of those neighbors and calculate them as predicted values. The similar characteristic of those neighbor is determined by distance indicators. There are two popular types of functions to calculate the distance between two vectors, Euclidean distance, and Manhattan distance. For this paper, Euclidean is used for distance calculation. It is also called by 2-norm. The equation below is used in the implementation to find the distance between neighbors.

$$f(x,y) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

So, it calculates the distance between the instance label and the training instance. These distances are sorted in ascending order, and the top-k is selected as neighbors. For classification, it uses plurality vote to determine the class. It takes the class that is found the most in the neighborhood. For regression, it uses Gaussian Kernel to smooth the output. The equation below is used to compute the weight values for the regression.

$$K(x_g,\ x^t) \ = \ exp[-\tfrac{1}{2\sigma}D(x_g,\ x^t)]$$

The equation below is Gaussian kernel smoother. It averages the k nearest neighbor to each data point to predict the class. So, it predicts the class value by total sum of weight and neighbor and divided by total sum of weight.

$$\hat{g}(x_g) \ = \ \frac{\sum_{t=1}^{n} K(\frac{x_g - x^t}{h}) r^t}{\sum_{t=1}^{n} K(\frac{x_g - x^t}{h})}$$

2.2 Edited K Nearest Neighbors

Edited KNN is very similar to standard KNN, but the goal of the edited KNN is removing the training data that doesn't help or improving to make a prediction. It uses step-wise backward elimination to reduce the data. In this algorithm, each of the training points is classified by iterating all of the other data points in the data to compute the prediction using standard KNN. When it predicted with standard KNN, it used 1 for nearest neighbor as the non-parametric estimator Alpaydın (2020). The reference refers to condensed NN, but it applied to edited KNN to make it fair between edited and condensed. If the prediction of the class disagrees, it drops the data from the training set. This process is repeated as long as the performance of the prediction continues to improve. So, It returns the reduced prediction.

2.3 Condensed K Nearest Neighbors

Condensed KNN is very similar to edited KNN. The goal of the Condensed KNN is it removes all the noise and errors in the dataset. It uses stepwise forward selection. In this algorithm, it starts from the empty set Z and passes over the instance in X in random order Alpaydın (2020). The first data point is added to the empty set. For each remaining point, it iterates through all points in the data to compute the distance using standard KNN. When it predicts with standard KNN, it finds the point that is the minimum distance to each point. So, the number of the nearest neighbor is set to 1. If the prediction of the point disagrees, it is added to Z. It repeats the process until there is no change. So, It returns the reduced prediction.

3. Experimental Approach

3.1 Cross Validation

For classification, it used 5-fold cross-validation and applied the stratification just like the previous assignment. For regression, it used 5-fold cross-validation, but stratification doesn't apply. Instead of stratification, it samples uniformly across all of the response values. It sorted the data on the predictor values and take every fifth point for a given fold.

3.2 Tuning

In this algorithm, several parameter factors need to consider how well the various KNN algorithm will perform. For classification, there is only one parameter that needs to consider, which is k. K is the number of neighbors that participate in the classification. For regression, there is three parameter that needs to consider, k, σ, ϵ. Sigma (σ) is the spread of value for Gaussian Kernel. Epsilon (ϵ) is the error threshold. The prediction needs to be within the error threshold.

These parameter can be arrange and edited as needed. For this implementation, the k is ranges from 1 to 10. The test values for sigma is sets as $\{0.01,\ 0.05,\ 0.1,\ 0.5,\ 1, 2\}$. For error threshold ($\epsilon$), it sets as $\{0.01,\ 0.05,\ 0.1,\ 0.5,\ 1\}$.

To tune these parameters, the total dataset is split into training (80%) and tuning (20%). The 5-fold cross-validation is applied to the tuning dataset to find the best combination of these parameters. For classification, it finds the best parameters that have maximum accuracy scores. For regression, it finds the best parameters that have minimum mean squared error. Once it finds the best parameter, it is applied to the training set. With the best parameter values, the training set is applied to 5-fold cross-validation. It applied either stratification or uniform sampling based on the types of dataset tasks.

4. Results of Experiments

The first table of each of the datasets demonstrated the best parameters and accuracy score/mean squared error for each algorithm. The second table of each of the datasets demonstrated the percent of the remaining dataset after edited and condensed KNN.

4.1 Breast Cancer Dataset

The Breast Cancer set represents a classification problem used to distinguish whether it is benign or malignant based on 10 feature attributes. Table 1 is demonstrated the accuracy score for each algorithm. Table 2 is demonstrated the percent of the remaining dataset after edited and condensed KNN.

Table 1: Accuracy Score of various KNN algorithm

	k	1	2	3	4	5	Avg
KNN	5	87.5%	98.2%	94.59%	98.2%	99.1%	95.52%
Edited KNN	5	86.61%	98.2%	94.59%	97.3%	98.2%	94.98%
Condensed KNN	4	87.5%	97.3%	92.79%	96.4%	97.3%	94.26%

Table 2: Percent of remaining dataset after Edited and Condensed KNN

	1	2	3	4	5	Avg
Edited KNN	96.62%	95.28%	96.63%	94.38%	94.61%	95.5%
Condensed KNN	10.81%	13.48%	12.81%	14.83%	15.73%	13.53%

4.2 Car Evaluation

The Car Evaluation set represents a classification problem used to distinguish car acceptability based on 6 feature attributes. Table 3 is demonstrated the accuracy score for each algorithm. Table 4 is demonstrated the percent of the remaining dataset after edited and condensed KNN.

Table 3: Accuracy Score of various KNN algorithm

	k	1	2	3	4	5	Avg
KNN	3	87.36%	85.51%	83.33%	87.68%	75.36%	83.85%
Edited KNN	3	87.36%	86.59%	84.42%	84.06%	74.64%	83.41%
Condensed KNN	3	86.28%	84.78%	85.51%	82.97%	75.72%	83.05%

Table 4: Percent of remaining dataset after Edited and Condensed KNN

	1	2	3	4	5	Avg
Edited KNN	83.06%	75.93%	77.83%	73.12%	87.51%	79.49%
Condensed KNN	63.13%	55.2%	48.14%	49.95%	46.15%	52.51%

4.3 Congressional Vote

The Congressional Vote set represents a classification problem used to distinguish house party (republican/democrat) based on 16 features of attributes of legislation votes. Table 5 is demonstrated the accuracy score for each algorithm. Table 6 is demonstrated the percent of the remaining dataset after edited and condensed KNN.

Table 5: Accuracy Score of various KNN algorithm

	k	1	2	3	4	5	Avg
KNN	9	97.14%	91.43%	91.43%	92.86%	100%	94.57%
Edited KNN	3	97.14%	94.29%	92.86%	92.86%	100%	95.43%
Condensed KNN	5	94.29%	92.86%	92.86%	94.29%	92.65%	93.39%

Table 6: Percent of remaining dataset after Edited and Condensed KNN

	1	2	3	4	5	Avg
Edited KNN	96.04%	96.04%	96.76%	97.48%	95%	96.26%
Condensed KNN	12.59%	14.39%	14.75%	10.07%	12.5%	13.0%

4.4 Abalone

The Abalone set represents a regression problem used to distinguish the associate age of abalone based on 8 features attributes. Table 7 is demonstrated the mean squared error for each algorithm. Table 8 is demonstrated the percent of the remaining dataset after edited and condensed KNN.

Table 7: MSE of various KNN algorithm

	k	σ	ϵ	1	2	3	4	5	Avg
KNN	5	2	-	8.782	4.018	4.063	4.965	32.583	10.882
Edited KNN	10	0.1	1	7.792	3.107	2.808	3.141	31.294	9.628
Condensed KNN	8	2	0.05	8.701	3.746	3.855	3.915	33.205	10.685

Table 8: Percent of remaining dataset after Edited and Condensed KNN

	1	2	3	4	5	Avg
Edited KNN	27.61%	28.99%	28.17%	29.77%	36.37%	30.18%
Condensed KNN	87.58%	88.4%	88.78%	87.77%	85.2%	87.55%

4.5 Computer Hardware

The Computer hardware set represents a regression problem used to distinguish the CPU performance based on 10 features attributes. Table 9 is demonstrated the mean squared error for each algorithm. Table 10 is demonstrated the percent of the remaining dataset after edited and condensed KNN.

Table 9: MSE of various KNN algorithm

	k	σ	ϵ	1	2	3	4	5	Avg
KNN	9	2	-	0.798	0.155	0.308	0.274	5.873	1.481
Edited KNN	10	1	0.01	0.828	0.146	0.464	0.252	6.5	1.638
Condensed KNN	1	1	0.05	0.764	0.235	0.294	0.575	5.437	1.461

Table 10: Percent of remaining dataset after Edited and Condensed KNN

	1	2	3	4	5	Avg
Edited KNN	72.18%	61.65%	66.92%	65.67%	79.26%	69.14%
Condensed KNN	42.86%	65.41%	60.9%	55.22%	47.41%	54.36%

4.6 Forest Fires

The Forest Fires set represents a regression problem used to predict the burned area of forest fires by using 12 attributes features of meteorological data. Table 11 is demonstrated the mean squared error for each algorithm. Table 12 is demonstrated the percent of the remaining dataset after edited and condensed KNN.

Table 11: MSE of various KNN algorithm

	k	σ	ϵ	1	2	3	4	5	Avg
KNN	8	0.01	-	2.185	2.27	1.303	0.899	8.109	2.953
Edited KNN	6	0.01	0.05	0.492	0.298	0.329	2.732	10.49	2.868
Condensed KNN	6	0.05	1	1.593	1.938	0.965	1.138	8.543	2.836

Table 12: Percent of remaining dataset after Edited and Condensed KNN

	1	2	3	4	5	Avg
Edited KNN	19.45%	17.93%	29.09%	40%	44.85%	30.26%
Condensed KNN	66.57%	69.91%	70%	53.94%	48.79%	61.84%

5. Behavior of Algorithms

Unlike other KNN algorithms, Condensed KNN passes over the instances in a random order Alpaydın (2020). So, the result of the condensed KNN is slightly changing every time I implemented the algorithm.

The performance of the various KNN algorithm tasks was not expected. Hypothesis 1 was the standard KNN will outperform than edited and condensed KNN. Throughout the 6 datasets (3 classifications, 3 regression), two datasets match with hypothesis 1. For classification, standard KNN outperformed edited and condensed KNN. In the congressional vote dataset, Edited KNN outperformed other KNN algorithms. But, it is very interesting that the condensed KNN showed poor performance in all classification tasks.

For regression, it showed a different result than classification. On the computer dataset, it showed Condensed KNN performed best, and Edited KNN performed worst. For the Abalone dataset, edited KNN outperformed compared to the other two algorithms. For Forest, condensed KNN outperformed. For both abalone and forest dataset, it is unexpected that standard KNN performed worst compared to other KNN algorithms.

Both edited and condensed KNN algorithms did a great job of reducing the dataset, but the execution time for the Edited and Condensed KNN algorithm was not expected at all. Hypothesis 2 was the execution time for both edited and condensed KNN will be faster than standard KNN because it reduced the dataset. During the implementation and tuning process, I found that execution time for edited and condensed KNN was very slow especially when the dataset was large (ex. Abalone). For standard KNN, the execution time was around 1 2 minutes to finish the tuning process. However, the execution time for edited and condensed KNN was around 6 20 minutes. For the large dataset, the abalone dataset, it took 2 and half hours to finish the tuning process. Even though the edited and condensed KNN reduced the compute time for each point in the training set, it is very interesting that reducing the training set doesn't result in reducing the execution time for the entire process.

For edited KNN, it reduced the training set by 10∼20% for classification and 50∼70% for regression. For condensed KNN, it reduced the training set by 50∼80% for classification and 20∼50% for regression. Even though the training set is reduced, it is also interesting that all of the performance scores were roughly in line with each other for classification and regression.

6. Conclusion

This paper implemented the various K-nearest neighbor algorithm: Standard KNN, Edited KNN, and Condensed KNN. Throughout the paper, I was able to learn how the various KNN algorithm tries to improve the performance by reducing data.

For Future analysis and implementation, it should implement an efficient algorithm that runs faster for the large dataset.

References

Ethem Alpaydın. *Introduction to Machine Learning, fourth edition.* The MIT Press, 2020.