CS205: Introduction to Artificial Intelligence, Dr. Eamonn Keogh

Project 2: Feature Selection with Nearest Neighbor

Name: Sourav Singha
SID: 862323554
Email Id: ssing263@ucr.edu
Date: June 10th 2022

In completing this assignment, I consulted:

- The CS 172 course recordings and CS 205 slides by Prof. Dr. E Keogh and notes annotated from lecture.
- Python 3.9 docs from
  https://docs.python.org/3.9/
- Project 2 guidelines Dr. Eamonn Keogh for report structure and outline
- Numpy python documentation
  https://numpy.org/doc/
- Matplotlib python documentation
  https://matplotlib.org/stable/index.html

The source code and algorithms project uses are an original work and developed solely for fulfilling the requirement of CS 205 Artificial Intelligence Project 1 submission. The utility functions and logic that are referred to are the following:

- Python Numpy subroutines for basic mathematical operations like square, sum, square root and slicing.
- Project 2 guidelines Dr. Eamonn Keogh for function and variable naming conventions, and test cases.

Outline

Report Summary

Introduction

The Nearest Neighbor algorithm which is used for find the similarity of a record between an already existing database or samples which has been recorded before. It is also mostly used to identify class of a unseen object by identifying its similarity to other observation of which the class is not from before. But its performance and accuracy are greatly affected by the choice of features we choose for calculating its distance.

The forward and backward selection are two approaches for selecting the optimal set of features for Nearest Neighbor algorithm which are as discussed below

1. **Forward selection:** The algorithm starts from an empty set and keeps on expanding by using a greedy approach until it has included all the feature in its set and the result is the set for which best accuracy was observed. For expanding the current set at every stage, it selects the features when added to already selected feature in set gives the highest accuracy until it has added all the features to its set.

2. **Backward Elimination:** The algorithm work in opposite manner to the it starts out with set containing all the features and keeps on removing an item form the set by which it observes the highest accuracy at every step until there is only one feature is remaining in the current set. The features for which highest accuracy has been observed is returned as the result.

Design and Working

The command line interface allows for easy access to the application and the user can enter the location to the dataset to be processed and then type of algorithm to use on it for the getting the optimal features, which are the Forward and Backward Selection approaches.

For deriving the accuracy of the current running feature set we use leave-one-out cross validation. It popular but computationally expensive process and is a type of the k-fold cross validation where the number of folds equals to the number of instances in the data set Here. We check the accuracy for every data point in the dataset, then by averaging the error we get the overall accuracy measure.

The design approach followed for the development followed a Functional Programming approach. The implementation can be broken down into these 4 parts:

1. **Driver function**: For interacting with the user though the Command Line Interface, reading the dataset, and calling the appropriate function for feature selection algorithm (Forward Selection / Backward Elimination) based on user input. It also further calculates the total time consumed.
2. **Forward Selection:** Performs the forward selection algorithm and utilizes the Get Accuracy function to get the accuracy for current running set.
3. **Backward Elimination:** Performs the forward selection algorithm and utilizes the Get Accuracy function to get the accuracy for current running set.
4. **Get Accuracy:** Performs 1-fold cross validation on the dataset using the group of the features passed to it. It utilizes Euclidean distance measure for performing the Nearest Neighbor algorithm.

Euclidean Distance Measure = $\sqrt{\sum_{i=1}^{n}(P_i - T)^2}$

Where, n : Number of Obs in the Training Set

P_i : i-th Feature Vector in the training Set

T : Feature vector of the 1 fold cross validation test rows

The implementation also makes use of multidimensional numpy arrays and avoid usage of loops in the leave-one-out cross validation to increase the performance and computational overhead, which has enabled to execute any of the test sets within a 1-minute mark.

Observations

The dataset in focus were the following:

1. Small dataset: CS205_SP_2022_SMALLtestdata__62.txt
2. Large dataset: CS205_SP_2022_Largetestdata__80.txt

The following values were observed for the forward and backward elimination while running the algorithms for finding the best features

1. Result Forward selection on Small dataset is {6, 9} and the accuracy is for it is 95.00 %
2. Result Backward Elimination on Small dataset is {6, 9} and the accuracy is for it is 95.00 %
3. Result Forward selection on Large dataset is {10, 36} and the accuracy is for it is 98.20 %
4. Result Backward Elimination on Large dataset is {10, 21} and the accuracy is for it is 85.30 %

Small Dataset

Step	Features	Accuracy (%)
1	{9}	82.33
2	{6, 9}	95.00
3	{6, 9, 10}	92.67
4	{2, 6, 9, 10}	89.00
5	{2, 6, 7, 9, 10}	83.67
6	{2, 3, 6, 7, 9, 10}	84.00
7	{2, 3, 6, 7, 8, 9, 10}	78.67
8	{1, 2, 3, 6, 7, 8, 9, 10}	78.00
9	{1, 2, 3, 4, 6, 7, 8, 9, 10}	73.33
10	{1, 2, 3, 4, 5, 6, 7, 8, 9, 10}	69.00

Step	Features	Accuracy (%)
1	{1, 2, 3, 4, 5, 6, 7, 8, 9, 10}	69.00
2	{1, 3, 4, 5, 6, 7, 8, 9, 10}	68.33
3	{1, 3, 4, 5, 6, 8, 9, 10}	68.67
4	{1, 3, 5, 6, 8, 9, 10}	72.33
5	{3, 5, 6, 8, 9, 10}	79.67
6	{3, 6, 8, 9, 10}	84.00
7	{6, 8, 9, 10}	85.67
8	{6, 9, 10}	90.33
9	{6, 9}	95.00
10	{9}	69.33

Table 1, 2: Small dataset: Forward Selection (left) and Backward Elimination (right) progress

The small dataset started with feature 9 which had the highest accuracy among features at 82.33 percent in forward selection and achieved the highest accuracy at stage 2 with feature set {6, 9}, Following which even by choosing the best features to add at every stage it kept on decreasing.

Where as the backward elimination started with all the features at accuracy 69.33 percent. However, it took quite long to find the optimal solution until the 9th elimination stage.

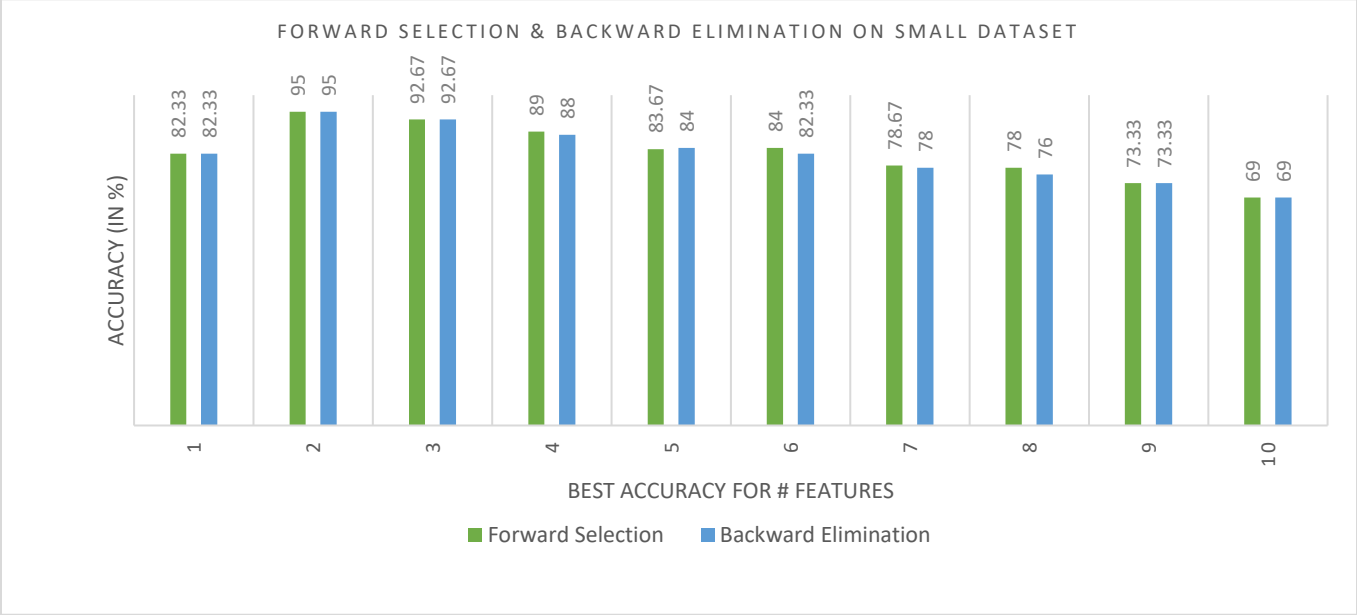

Figure 1: Number of Features vs best Accuracy for Small Dataset at every expanding and elimination stage

The highest accuracy was observed when the features set was of size 2, combination of features more lowered the accuracy for the small dataset monotonically.

Large Dataset

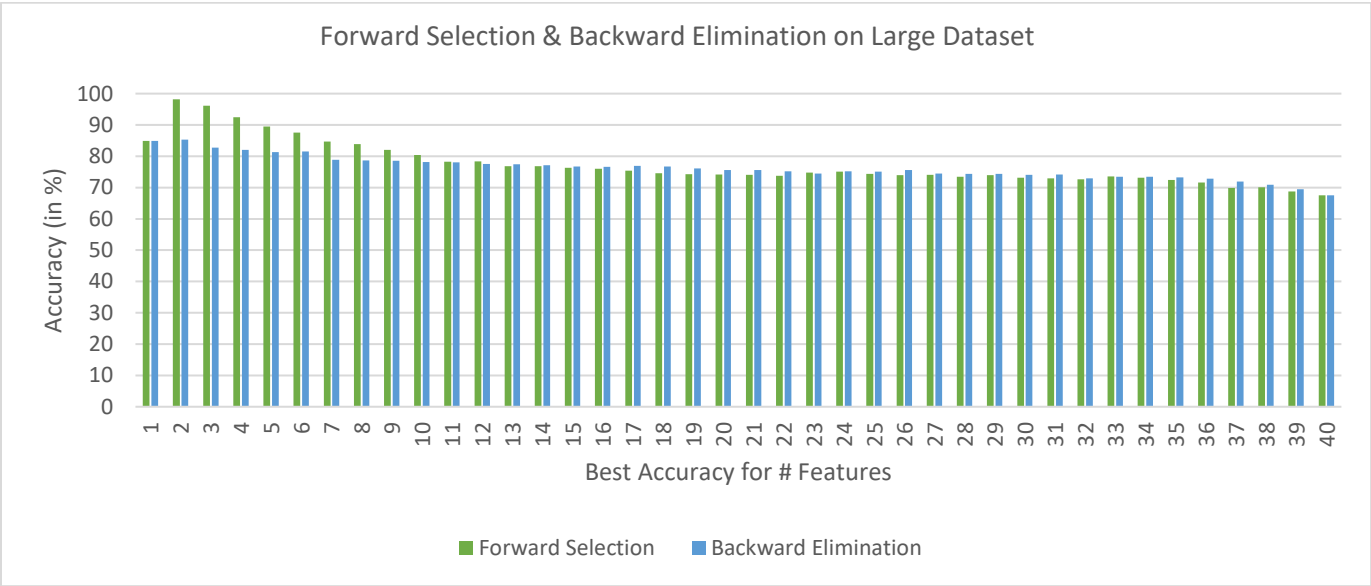

Figure 2: Number of Features vs Accuracy for Large Dataset in Forward Selection and backward Elimination

The highest accuracy was observed when the features set was of size 2, combination of more features slowly lowered the accuracy for the large dataset in a similar manner to the observations from small dataset. The best set of features was {10, 36} and the corresponding accuracy is 98.20 % which was found by forward elimination.

The forward selection was started by selecting 10 then 36, and subsequently adding in the order of 26, 4, 10 … as such.

While the backward elimination started with the entire full set and removed features in the following order 19, 36, 34, 13 …

Result of Backward Elimination on Large dataset is {10, 21} and the corresponding accuracy is 85.30 %. The forward selection algorithm also further provided feature sets with higher accuracy for given size than backward elimination till the size was 12. After which the backward elimination generated sets with higher accuracy for the same feature set size.

System Configuration

The forward selection and backward elimination on the datasets have been performed on a windows system with the following configuration:

Hardware Configuration:

- CPU: AMD Ryzen 4600H (6 core, 3.8GHz)
- RAM: 16 GB DDR4s
- Graphics Card: NVIDIA GTX 1650 (Not used for this project)

Software/Programming Language:

- Programming Language: Python == 3.9
- Libraries: Numpy == 1.22, time

1. For known difficulty levels

Dataset	Forward Selection (secs)	Backward Selection (secs)
CS205_SP_2022_SMALLtestdata__62.txt (small dataset)	0.44	0.47
CS205_SP_2022_Largetestdata__80.txt (Large dataset)	38.87	43.14

Table 1: Execution times for the datasets

Conclusion

The results of the project help can be summarized by the following points as listed below:

1. The optimal features for Nearest Neighbor algorithm for determining the feature class can be achieved just by selecting appropriate features which requires around selection of two feature out of the 10 or 40 features.
2. Most of the other features or selection of all features were less effective in determining the class.

3. For the small dataset the highest accuracy was observed at 95 % by both forward selection and backward elimination using the features {6, 9}.
4. For the large dataset the highest accuracy was observed at 98 % by forward selection and the feature set discovered was 10, 36.
5. It can also be concluded that forward selection and backward elimination may not give the same resultant optimal feature set as the algorithms is not guaranteed to reach the global maxima equally.

Sample trace

Forward Selection

Dataset Used: CS205_SP_2022_SMALLtestdata__62.txt (Small Dataset)

Output:

```
Welcome to Sourav Singha Feature Search Algorithm

Enter name of the dataset you want to process: ./Data/CS205_SP_2022_SMALLtestdata__62.txt
Type the number of the algorithm you want to run.
       1) Forward Selection
       2) Backward Elimination
Enter your choice: 1
Beginning Search.

Using features(s) {1} accuracy is 64.00 %
Using features(s) {2} accuracy is 60.00 %
Using features(s) {3} accuracy is 62.00 %
Using features(s) {4} accuracy is 64.33 %
Using features(s) {5} accuracy is 64.33 %
Using features(s) {6} accuracy is 69.33 %
Using features(s) {7} accuracy is 66.00 %
Using features(s) {8} accuracy is 67.00 %
Using features(s) {9} accuracy is 82.33 %
Using features(s) {10} accuracy is 60.33 %

Feature set {9} was best, accuracy is 82.33 %

Using features(s) {1, 9} accuracy is 83.00 %
Using features(s) {2, 9} accuracy is 84.67 %

[Omitting to save space]


Feature set {1, 2, 3, 4, 6, 7, 8, 9, 10} was best, accuracy is 73.33 %

(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Using features(s) {1, 2, 3, 4, 5, 6, 7, 8, 9, 10} accuracy is 69.00 %

Feature set {1, 2, 3, 4, 5, 6, 7, 8, 9, 10} was best, accuracy is 69.00 %

(Warning, Accuracy has decreased! Continuing search in case of local maxima)

Finished search!! The best feature subset is {6, 9}, which has an accuracy of 95.00 %

Execution Time: 0.44 seconds

Process finished with exit code 0
```

Backward Selection

Dataset Used: CS205_SP_2022_SMALLtestdata__62.txt (Small Dataset)

```
Welcome to Sourav Singha Feature Search Algorithm

Enter name of the dataset you want to process: ./Data/CS205_SP_2022_SMALLtestdata__62.txt
Type the number of the algorithm you want to run.
       1) Forward Selection
       2) Backward Elimination
Enter your choice: 2
Beginning Search.

Using features(s) {1, 2, 3, 4, 5, 6, 7, 8, 9, 10} accuracy is 69.00 %

Feature set {1, 2, 3, 4, 5, 6, 7, 8, 9, 10} was best, accuracy is 69.00 %

Using features(s) {2, 3, 4, 5, 6, 7, 8, 9, 10} accuracy is 71.00 %
Using features(s) {1, 3, 4, 5, 6, 7, 8, 9, 10} accuracy is 73.33 %
Using features(s) {1, 2, 4, 5, 6, 7, 8, 9, 10} accuracy is 70.33 %
Using features(s) {1, 2, 3, 5, 6, 7, 8, 9, 10} accuracy is 73.00 %
Using features(s) {1, 2, 3, 4, 6, 7, 8, 9, 10} accuracy is 73.33 %
Using features(s) {1, 2, 3, 4, 5, 7, 8, 9, 10} accuracy is 65.00 %
Using features(s) {1, 2, 3, 4, 5, 6, 8, 9, 10} accuracy is 72.67 %
Using features(s) {1, 2, 3, 4, 5, 6, 7, 9, 10} accuracy is 72.33 %
Using features(s) {1, 2, 3, 4, 5, 6, 7, 8, 10} accuracy is 62.67 %
Using features(s) {1, 2, 3, 4, 5, 6, 7, 8, 9} accuracy is 68.33 %

Feature set {1, 3, 4, 5, 6, 7, 8, 9, 10} was best, accuracy is 73.33 %

Using features(s) {3, 4, 5, 6, 7, 8, 9, 10} accuracy is 75.00 %
Using features(s) {1, 4, 5, 6, 7, 8, 9, 10} accuracy is 69.33 %
Using features(s) {1, 3, 5, 6, 7, 8, 9, 10} accuracy is 75.00 %
Using features(s) {1, 3, 4, 6, 7, 8, 9, 10} accuracy is 73.00 %
Using features(s) {1, 3, 4, 5, 7, 8, 9, 10} accuracy is 71.67 %
Using features(s) {1, 3, 4, 5, 6, 8, 9, 10} accuracy is 76.00 %
Using features(s) {1, 3, 4, 5, 6, 7, 9, 10} accuracy is 72.00 %
Using features(s) {1, 3, 4, 5, 6, 7, 8, 10} accuracy is 67.00 %
Using features(s) {1, 3, 4, 5, 6, 7, 8, 9} accuracy is 68.67 %

Feature set {1, 3, 4, 5, 6, 8, 9, 10} was best, accuracy is 76.00 %

[Omitting to save space]

Using features(s) {9, 10} accuracy is 81.67 %
Using features(s) {6, 10} accuracy is 72.67 %
Using features(s) {6, 9} accuracy is 95.00 %

Feature set {6, 9} was best, accuracy is 95.00 %

Using features(s) {9} accuracy is 82.33 %
Using features(s) {6} accuracy is 69.33 %

Feature set {9} was best, accuracy is 82.33 %

(Warning, Accuracy has decreased! Continuing search in case of local maxima)

Finished search!! The best feature subset is {6, 9}, which has an accuracy of 95.00 %

Execution Time: 0.47 seconds

Process finished with exit code 0
```

Source code

The source code for the developed and used in preparing the report can be found at

Github url: https://github.com/ssing263/CS-205-Artificial-Intelligence/blob/main/NearestNeighbour.py