CS170 Artificial Intelligence, Dr. Eamonn Keogh
Project 2

Minsoo Kim
SID 862238343
mkim410@ucr.edu
Date 12-08-2022

I consulted the following material while completing this assignment

- Lecture slides and recordings regarding nearest neighbor, project 2 briefing, and search algorithms
- Python 3.6 documentation
    - specifically the python3 library documentation
    - https://docs.python.org/3/library/
- Pandas documentation
    - https://pandas.pydata.org/docs/reference/frame.html
- Matlab documentation
    - https://www.mathworks.com/help/matlab/matlab-engine-for-python.html
- All the code written is original. I did not consult any written code elsewhere.
- All the diagrams have been produced by myself with the assistance of google spreadsheets
- Libraries used
    - ***time*** (used to measure the elapsed time running the algorithm)
    - ***math*** (used to calculate the euclidean distance)
    - ***pandas*** and ***matploblib.pyplot*** (used to creating bar charts)

Outline Report

- Cover Page (1)
- Report (2-6)
- Tracings (6-8)

Code

Note: You can see the full source code on my GitHub Repository
https://github.com/minsooerickim/feature_selection_with_nearest_neighbor

CS170 - 8 Puzzle Project

Minsoo Kim, SID 862238343, 12-08-2022

Introduction

Feature selection is one of the algorithms in machine learning. It is often used for classification problems as the process of selecting the best feature or features that classify an object. For instance, Feature selection can be used to solve the pollination problem shown in machine learning lecture part 4.

Forward Selection, Backwards Elimination using Nearest Neighbor on Small Data

Figure 1 shows the accuracies of the best feature subsets at each level when running forward selection with the nearest neighbor algorithm on one of the provided datasets, CS170_Small_Data_52.txt. This dataset contains 6 features with 500 instances.

At the very beginning of the search, the accuracy isn't all that bad at around 82.8% accuracy with no features. However, we can quickly see an increase in accuracy when we start adding features. Specifically, our accuracy jumps to 85.2% as soon as we add feature 5. Then, we hit the highest accuracy of 94.8% when we have 2 features,[5, 3], in our feature set. After the 2nd feature selection, we can see that adding more features only decreases our accuracy. We can notice that more features don't necessarily mean higher accuracy. In fact, with all 6 features, the accuracy is 81.6% which is actually less than having no features at all with 82.8% accuracy. Therefore, we can conclude that the best feature set for this specific data set is `[5, 3]` with an accuracy of `94.8%`.

Figure 1: Accuracy of different feature subsets using Forward Selection on small data

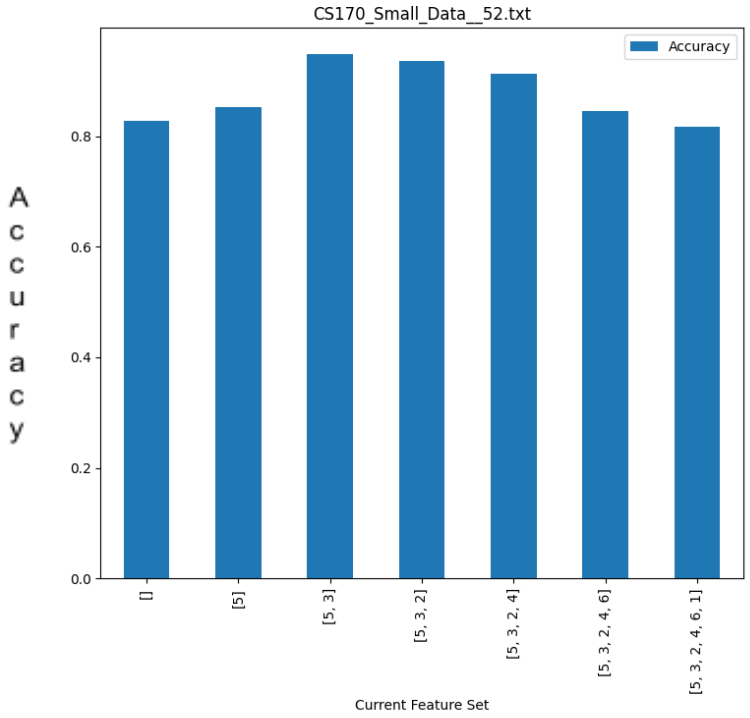
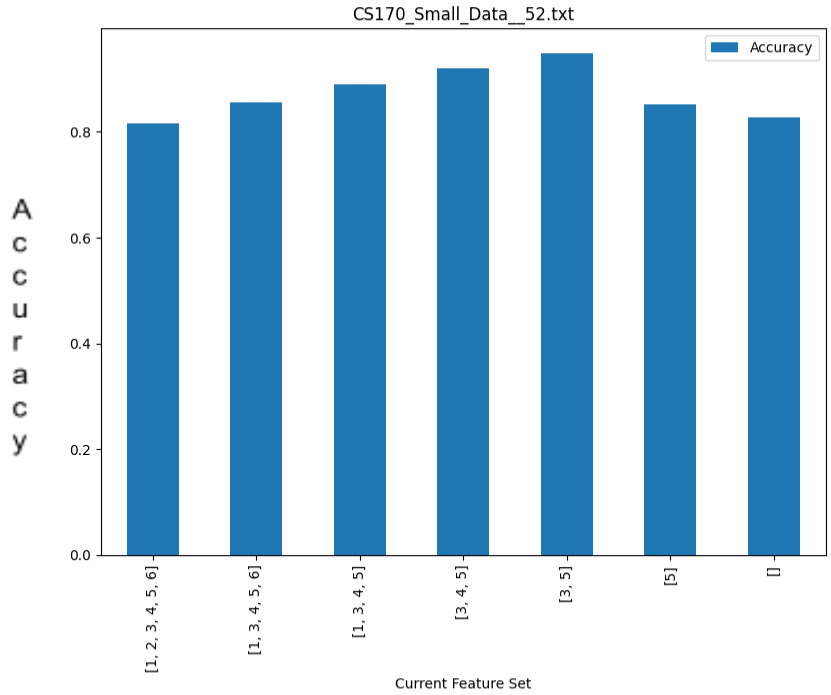


Figure 2 shows the accuracies of the best feature subsets at each level when running backward elimination with the nearest neighbor algorithm on the same small data set as

Figure 1. Backward selection starts with all of the features which gives an accuracy of 81.6%. As we remove one feature at a time level by level, we see an increase in accuracy, with the max accuracy being 94.8% with the feature subset of [3, 5]. We can see that the results are actually identical to that of the forward selection algorithm. From such results, we can verify that features 3 and 5 are very good features.

Figure 2: Accuracy of different feature subsets using Backwards Elimination on small data



Conclusion for the small data set

Looking at both figures 1 and 2, it is also worth noting that feature subsets [3,4,5] (92%) and [5,3,2](93.6%) are very close in accuracy with [3,5](94.8%) so it is possible that features 4 and 2 are also very good features.

Forward Selection, Backwards Elimination using Nearest Neighbor on Large Data

Figure 3 shows the accuracies of the best feature subsets at each level when running forward selection with the nearest neighbor algorithm on a large dataset, CS170_Large_Data__71.txt. This dataset contains 40 features with 1000 instances.

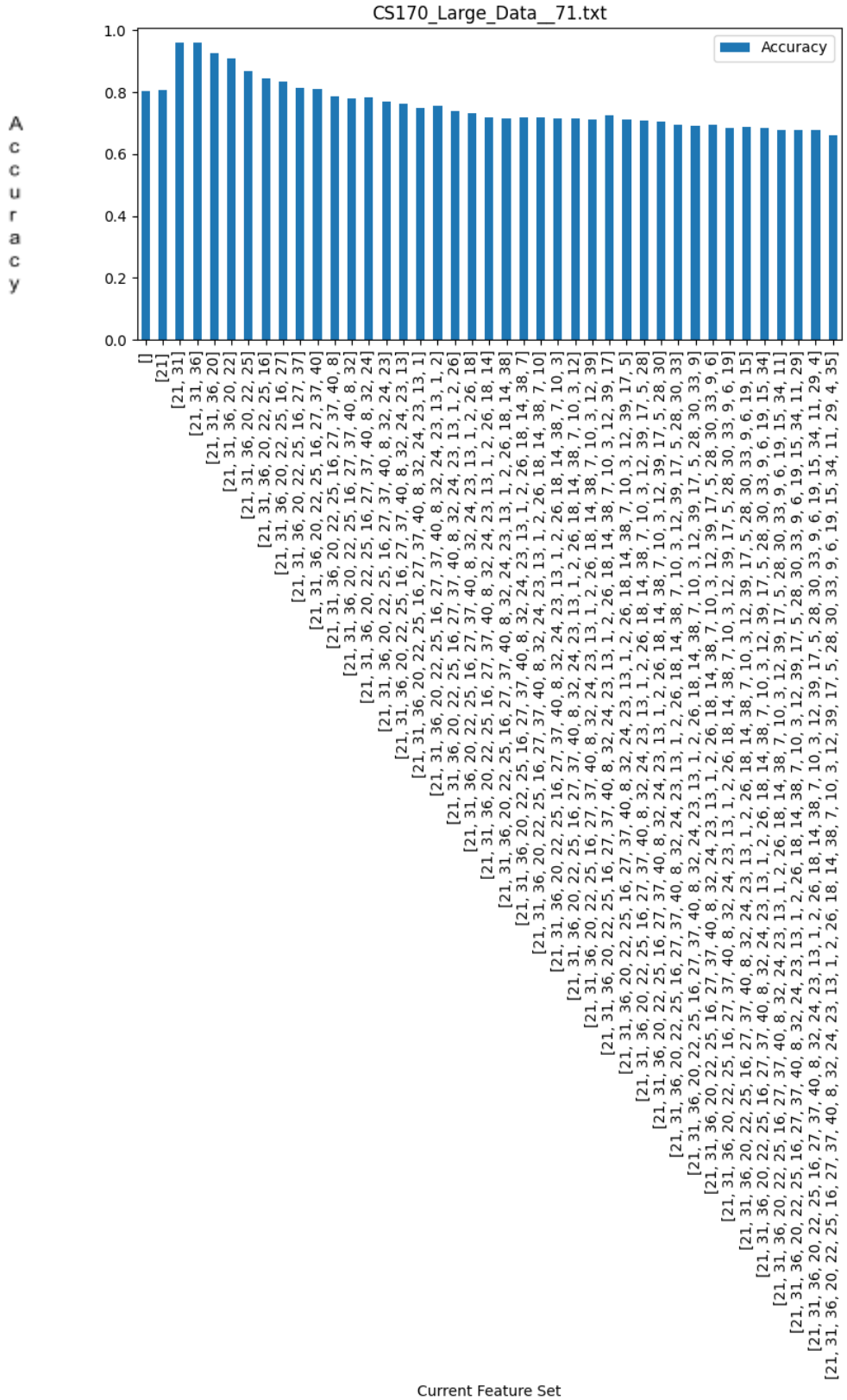


Figure 3: Accuracy of different feature subsets using Forward Selection on large data

Starting with the empty set in Figure 3, the accuracy is at 80.3%. We can observe the accuracy quickly increases as we start to add the best features level by level. In fact, the max accuracy of 96% is reached in the 3rd level with the subsets [21,31,36]. Knowing this is the highest accuracy, it would be very efficient to end the algorithm here. However, we have to continue searching in case there is a higher accuracy later on in the search. After the peak in the 3rd level of the search, the accuracy gradually decreases all the way down to the lowest point at 66.2% with all of the features considered.

Figure 4: Accuracy of different feature subsets using Backwards Elimination on large data

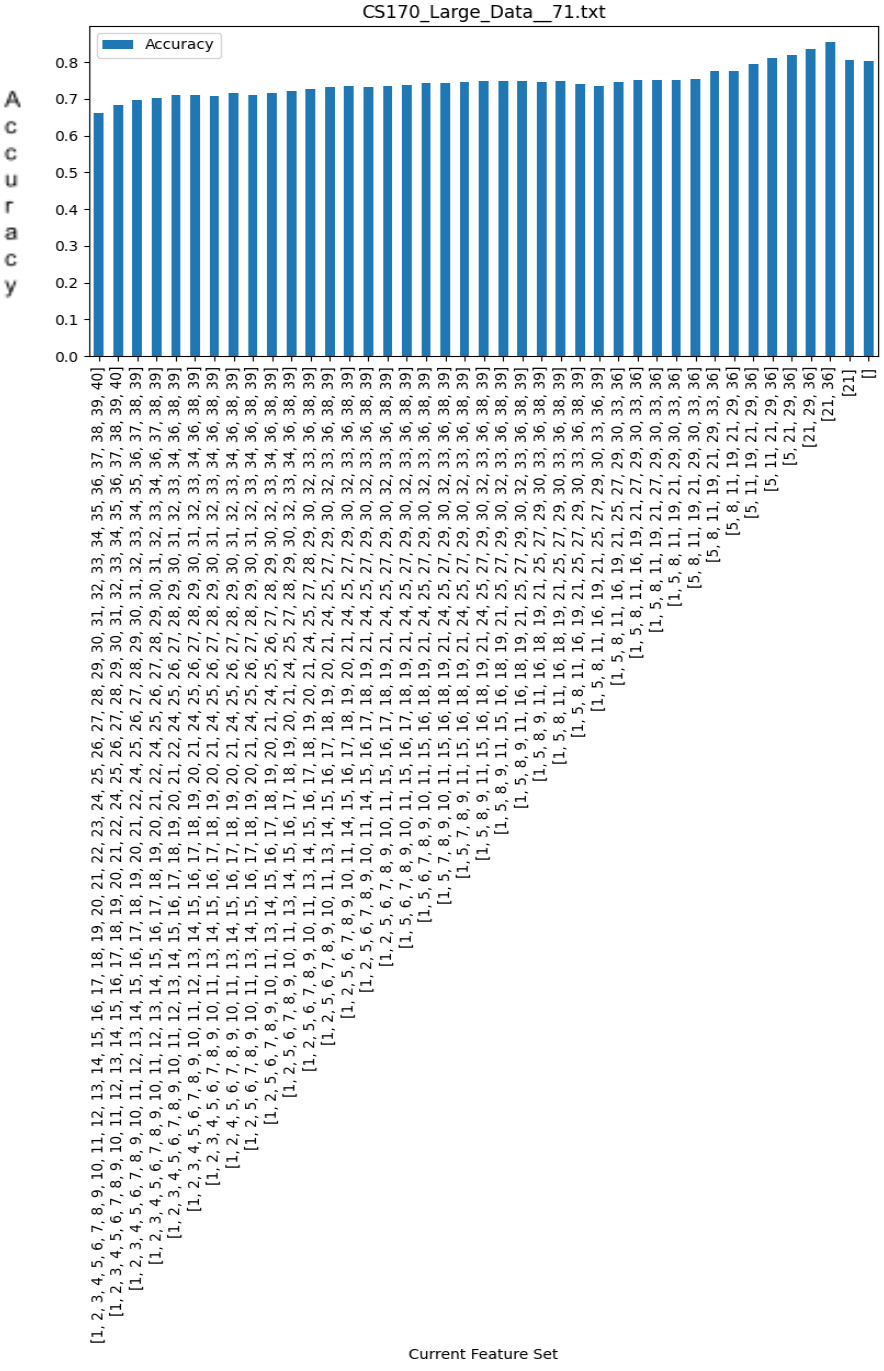


Figure 4 shows the accuracies of the best feature subsets at each level when running backward elimination with the nearest neighbor algorithm on the same data set as Figure 3. Unlike Figure 3, The highest accuracy is found towards the end of the search space. Therefore, for this specific dataset, we would have found the optimal solution much quicker using forward selection instead of backward elimination.

The results of backward elimination are very similar to that of the forward selection algorithm. Backward elimination found feature subsets [21, 36] to be the best at an accuracy of 85.6%. This ensures us that features [21, 36] are very good features since forward selection found [21,31,36] to be the best feature subset.

Conclusion for the large data set
While the backward elimination didn't catch feature 31 as a part of its best feature subset, we did see it in the forward selection with a very high accuracy of 96%. From such information, we can also conclude that feature 31 could also be a very good choice and might be chosen with backward selection paired with perhaps a different nearest-neighbor distance algorithm.

Computational effort for the search
I ran all of the computations including the ones described above on my laptop with an Apple Silicon M1 Chip and 16 GB of memory. Table 1 below shows the run time for the 4 searches performed which have been shown in the diagrams above.

Table 1: run time of the 4 searches I performed on 2 different data sets

	Small Data 52(6 features, 500 instances)	**Large Data 71(40 features, 1000 instances)**
Forward Selection	3.9 seconds	34.8 minutes
Backward Elimination	5.1 seconds	61.1 minutes

The runtime of the 4 searches algorithm vary approximately +- 10% from the times in table 1. However, the best feature subsets stayed consistent in all of the runs I performed. I did notice that backward elimination on average took longer than forward selection on both CS170_Small_Data__52.txt and CS170_Large_Data__71.txt.

Tracings
I have posted the tracings for all 4 searches I performed here, https://github.com/minsooerickim/feature_selection_with_nearest_neighbor/tree/main/traces, on GitHub.

Below is a tracing of the forward selection on the CS170_Small_Data__52.txt data set. I am ONLY showing 1 tracing on this report. More can be found on the link above.

```
This data set has 6 features with 500 instances.
```

```
Running nearest neighbor with all 6, using "leaving-one-out" evaluation, I get an accuracy
of 0.816

1. forward selection
2. backward_elimination

accuracy with 0 features, 0.828

On the 1 level of the search tree
-- Considering adding the 1 feature
---- accuracy with [1] : 0.692
-- Considering adding the 2 feature
---- accuracy with [2] : 0.712
-- Considering adding the 3 feature
---- accuracy with [3] : 0.74
-- Considering adding the 4 feature
---- accuracy with [4] : 0.716
-- Considering adding the 5 feature
---- accuracy with [5] : 0.852
-- Considering adding the 6 feature
---- accuracy with [6] : 0.706

feature subset [5] had the highest accuracy of 0.852
On level 1, I added feature 5 to current set
-----------------------------------------------

On the 2 level of the search tree
-- Considering adding the 1 feature
---- accuracy with [5, 1] : 0.816
-- Considering adding the 2 feature
---- accuracy with [5, 2] : 0.834
-- Considering adding the 3 feature
---- accuracy with [5, 3] : 0.948
-- Considering adding the 4 feature
---- accuracy with [5, 4] : 0.84
-- Considering adding the 6 feature
---- accuracy with [5, 6] : 0.786

feature subset [5, 3] had the highest accuracy of 0.948
On level 2, I added feature 3 to current set
-----------------------------------------------

On the 3 level of the search tree
-- Considering adding the 1 feature
---- accuracy with [5, 3, 1] : 0.916
-- Considering adding the 2 feature
---- accuracy with [5, 3, 2] : 0.936
-- Considering adding the 4 feature
---- accuracy with [5, 3, 4] : 0.92
-- Considering adding the 6 feature
---- accuracy with [5, 3, 6] : 0.924
```

```
The best accuracy at this level, [5, 3, 2] : 0.936 was less than the best so far accuracy of
0.948
On level 3, I added feature 2 to current set
-----------------------------------------------

On the 4 level of the search tree
-- Considering adding the 1 feature
---- accuracy with [5, 3, 2, 1] : 0.886
-- Considering adding the 4 feature
---- accuracy with [5, 3, 2, 4] : 0.912
-- Considering adding the 6 feature
---- accuracy with [5, 3, 2, 6] : 0.894

The best accuracy at this level, [5, 3, 2, 4] : 0.912 was less than the best so far accuracy
of 0.948
On level 4, I added feature 4 to current set
-----------------------------------------------

On the 5 level of the search tree
-- Considering adding the 1 feature
---- accuracy with [5, 3, 2, 4, 1] : 0.842
-- Considering adding the 6 feature
---- accuracy with [5, 3, 2, 4, 6] : 0.846

The best accuracy at this level, [5, 3, 2, 4, 6] : 0.846 was less than the best so far
accuracy of 0.948
On level 5, I added feature 6 to current set
-----------------------------------------------

On the 6 level of the search tree
-- Considering adding the 1 feature
---- accuracy with [5, 3, 2, 4, 6, 1] : 0.816

The best accuracy at this level, [5, 3, 2, 4, 6, 1] : 0.816 was less than the best so far
accuracy of 0.948
On level 6, I added feature 1 to current set
-----------------------------------------------

The best features are [5, 3] with accuracy 0.948

Elapsed time: 3.954 seconds
```

Here is the link to my original source code on GitHub,
https://github.com/minsooerickim/feature_selection_with_nearest_neighbor