CS-205. Introduction to Artificial Intelligence
Project 2
Amrutha Alewoor | Net ID: aalew002 | Student ID: 862395063
Akash Bilgi | Net ID: abilg003 | Student ID: 862395080

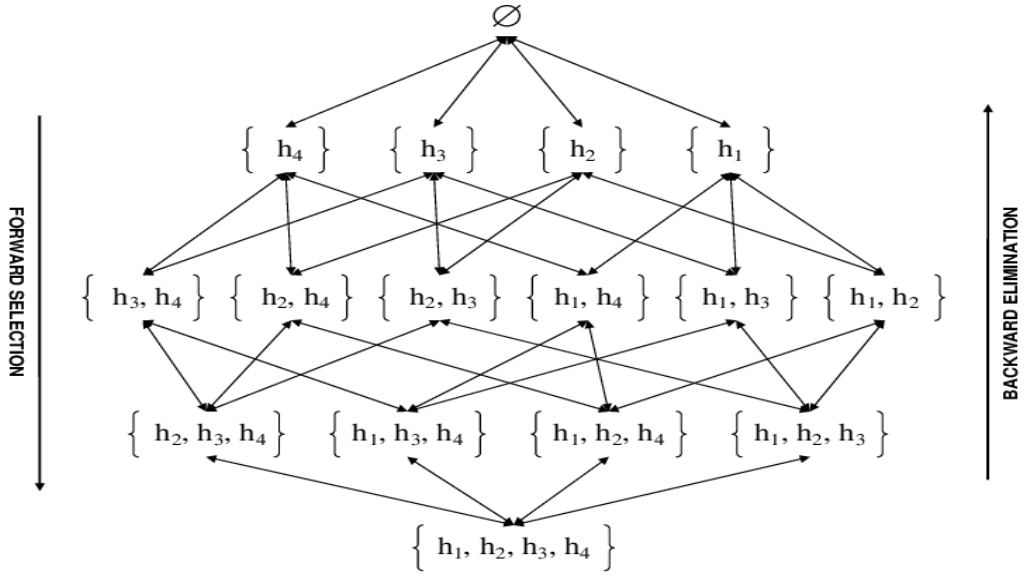
Introduction

The Nearest Neighbor Algorithm is a popular machine learning algorithm used for classification and regression tasks. It relies on the similarity between data points to make predictions. In this project, we focus on implementing a wrapper-based nearest neighbor algorithm for feature selection. Feature selection is a crucial step in machine learning that involves identifying the most relevant features from the original dataset to improve model performance and reduce computational complexity.

Feature selection methods can be categorized into filter methods, wrapper methods, and embedded methods. For our project, we concentrate on wrapper methods, specifically forward selection and backward elimination. These techniques iteratively add or remove features based on their impact on the model's performance.

Forward selection starts with an empty set of features and progressively adds one feature at a time. In each iteration, the algorithm assesses the performance of the model with the addition of each remaining feature and selects the one that contributes the most to accuracy improvement. This process continues until no further improvement is observed or a stopping criterion is met.

On the other hand, backward elimination begins with the full set of features and iteratively removes one feature at a time. In each iteration, the algorithm evaluates the model's performance without each remaining feature and eliminates the one that causes the least degradation. This process continues until no further improvement is observed or a stopping criterion is met.



Objective:

The main objective of our project is to implement the wrapper-based nearest neighbor algorithm for feature selection. We specifically focus on forward selection and backward elimination. By applying these techniques, we aim to identify the most relevant features that enhance classification accuracy.

To evaluate our implementation, we conduct experiments on three synthetic datasets provided to us. These datasets vary in size, ranging from small to large and extra large. Additionally, we also apply our feature selection algorithm to a real-world classification dataset obtained from www.kaggle.com. This real-world dataset allows us to assess the performance and practical applicability of our algorithm.

Through our project, we aim to demonstrate the effectiveness of our wrapper-based nearest neighbor algorithm for feature selection. By performing experiments on synthetic and real-world datasets, we seek to gain insights into the impact of feature selection on classification accuracy. Our project emphasizes the importance of selecting informative features to improve classification performance.

Feature Selection Methods

1. Leave-One-Out Cross-Validation:
   Leave-One-Out (LOO) cross-validation is a technique used for evaluating the performance of machine learning models. In LOO, each instance in the dataset is left out as a validation set, while the model is trained on the remaining instances. This process is repeated for each instance, and the performance of the model is averaged over all iterations. LOO is particularly useful when the dataset is small or when there is limited data available.

2. Forward Selection:
   Forward Selection is a feature selection method that starts with an empty set of features and iteratively adds the best-performing feature at each step. At each iteration, the algorithm evaluates the performance of all remaining candidate features and selects the one that improves the model's performance the most. This process continues until a stopping criterion is met, such as reaching a predefined number of features or observing a decrease in performance.

3. Backward Elimination:
   Backward Elimination is a feature selection method that starts with a full set of features and iteratively removes the least significant feature at each step. At each iteration, the algorithm evaluates the performance of the model with one feature removed and selects the feature whose removal causes the least decrease in performance. This process continues until a stopping criterion is met, such as reaching a predefined number of features or observing a decrease in performance.

4. Threshold for Early Abandoning:
   In the case of XXXL data, which represents extremely large datasets, a threshold of 0.01 is used for early abandoning. This means that if the improvement in accuracy achieved by adding or removing a feature is less than 0.01, the algorithm stops the selection process, assuming that further iterations are unlikely to yield significant improvements. This threshold helps to reduce the computational burden associated with processing large datasets while still allowing for effective feature selection.

Computing Accuracy for Feature set:

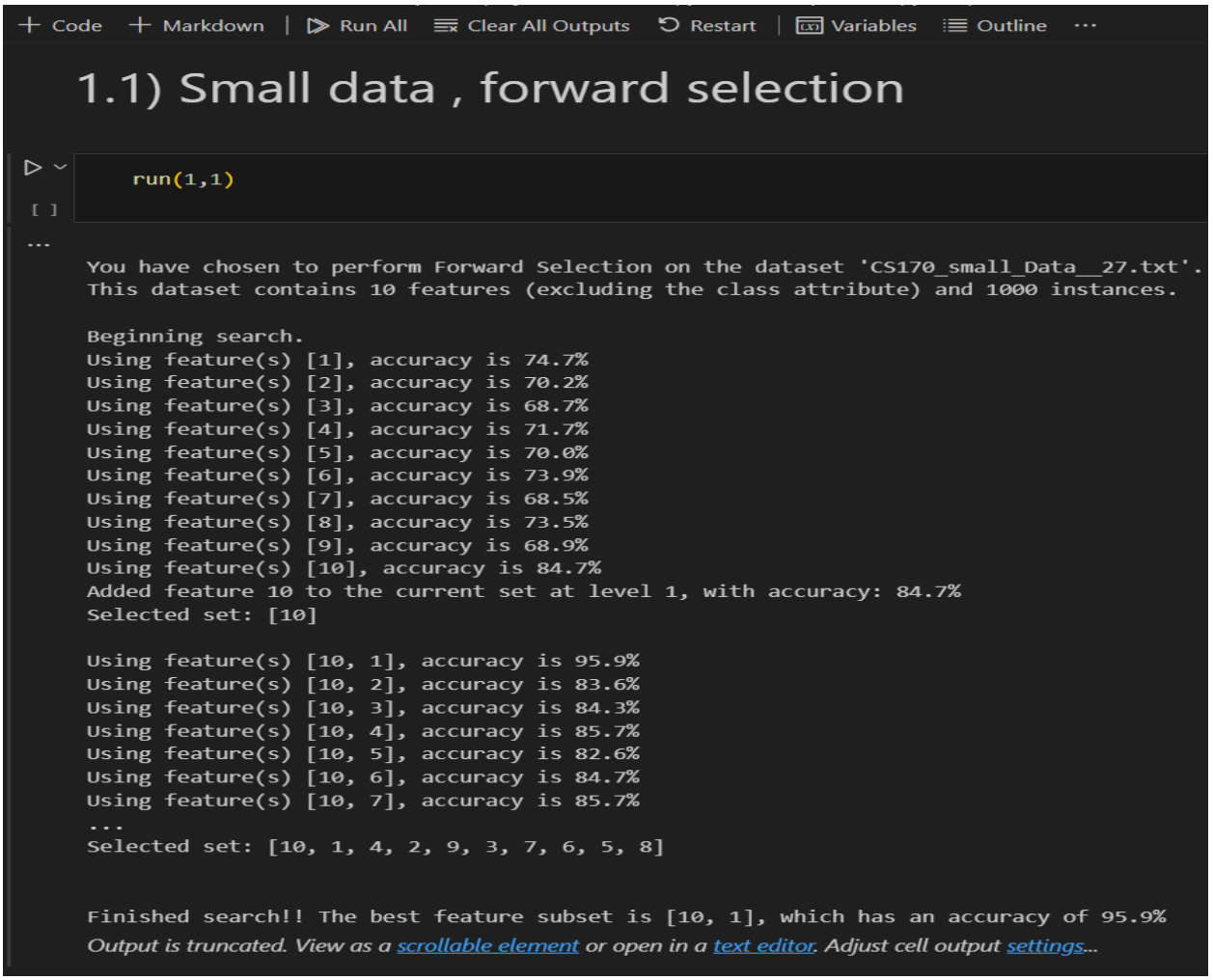
The accuracy is computed by comparing the predicted labels of the selected feature subsets with the actual labels in the dataset. It is a measure of how well the model is able to correctly classify instances. We use Euclidean distance to classify instances.

To calculate the accuracy, we divide the number of correctly classified instances by the total number of instances in the dataset. Mathematically, it can be expressed as:

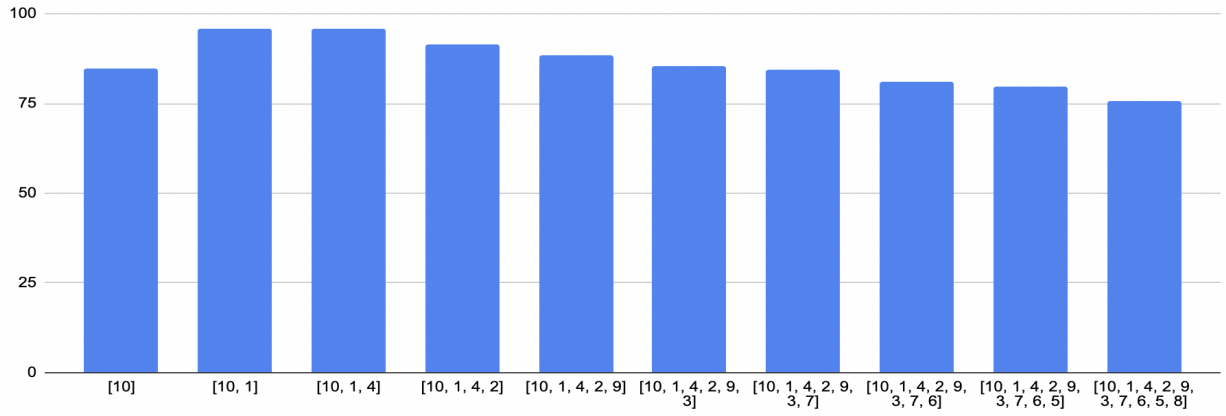
Accuracy = (Number of Correctly Classified Instances / Total Number of Instances) * 100

Small dataset: The dataset we have used for small is `CS170_small_Data__27.txt` Since the the birthday of the younger member is on 27. The dataset contains 10 features and 1000 instances, let us run all the combinations of features and check which one gives better results.

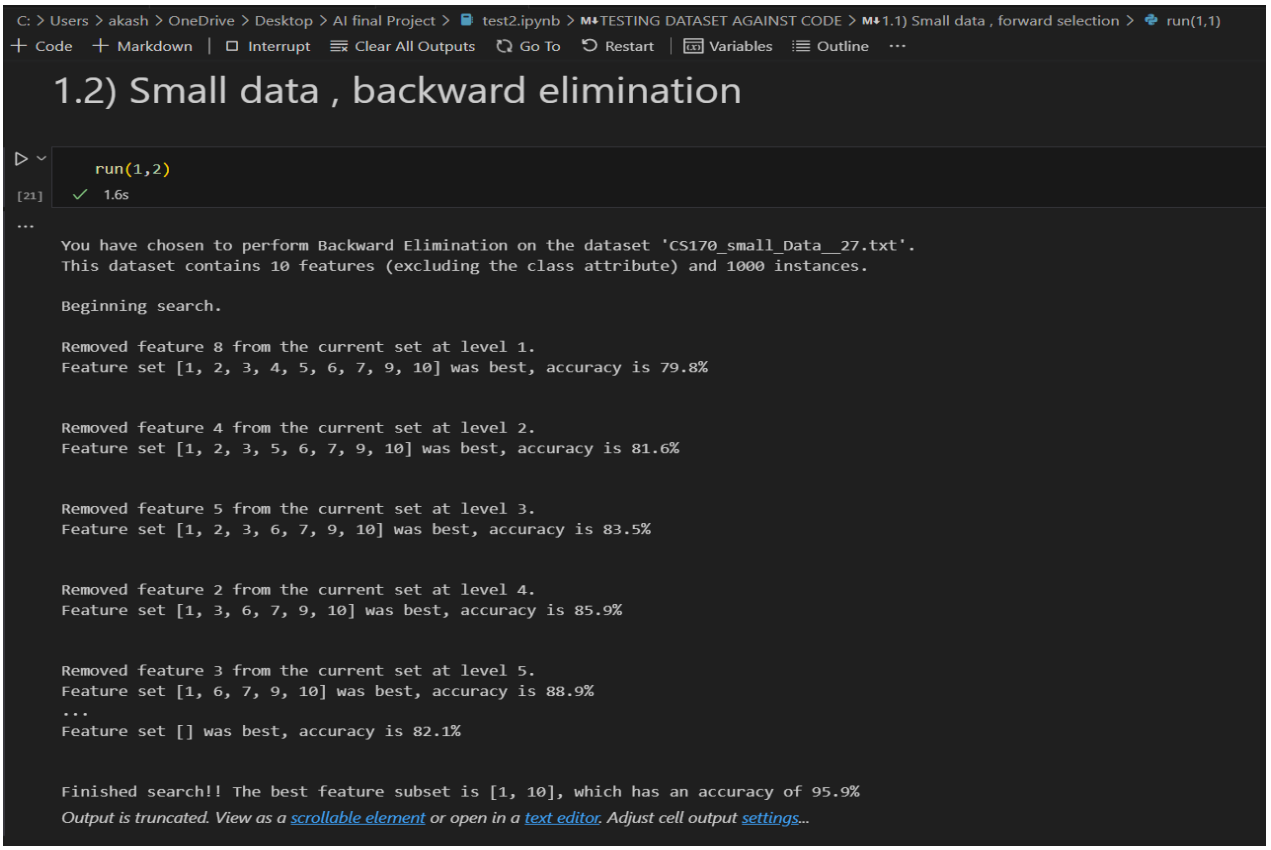
Forward Selection: In the forward selection process, we evaluated all possible combinations of features in the small dataset using the nearest neighbor algorithm. After considering each feature individually and in combination with other features, we found that the highest accuracy of 95.9% was achieved when using the feature set [10, 1]. This implies that the combination of features 1 and 10 together provides the best results when labeling an unknown instance.



The accuracy of 95.9% indicates that the selected features play a significant role in accurately classifying instances in the small dataset. By including features 1 and 10, we can effectively capture the underlying patterns and improve the predictive power of the nearest neighbor algorithm. Below graph visually represents the accuracy vs feature selection for Forward Selection.

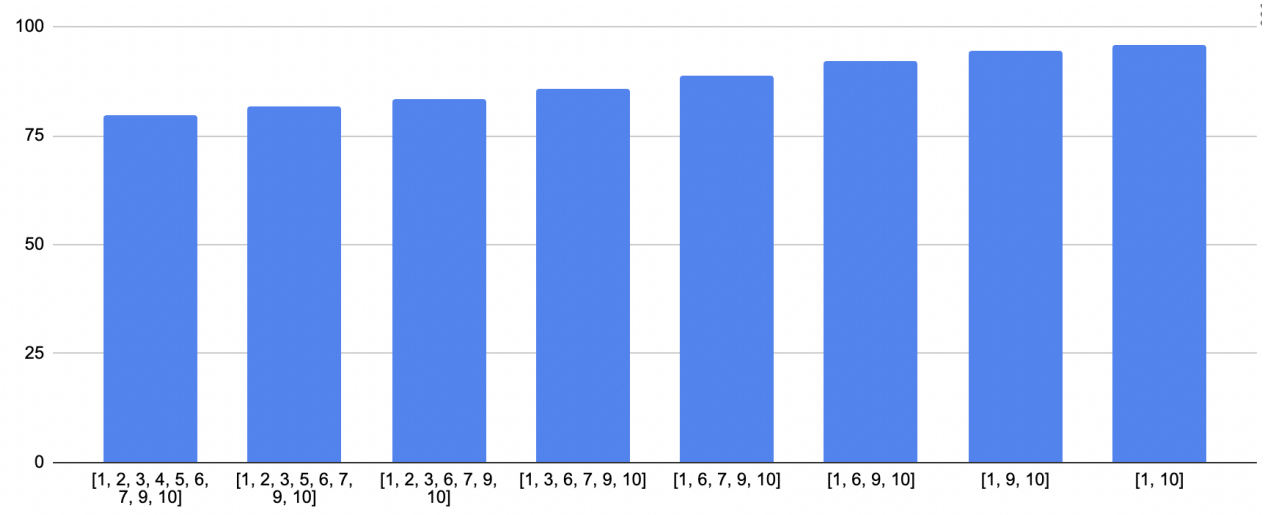


Backward Elimination: After eliminating undesirable combinations of features, we see that the highest accuracy is obtained for feature set [1 and 10] which is 95.9%. The consistency in results between forward selection and backward elimination, both yielding an accuracy of 95.9% with the same feature set, reinforces the importance of features 1 and 10 in accurately classifying instances. These features demonstrate their high informativeness and their significant contribution to improving the nearest neighbor algorithm's accuracy.
Below is the output for small dataset.



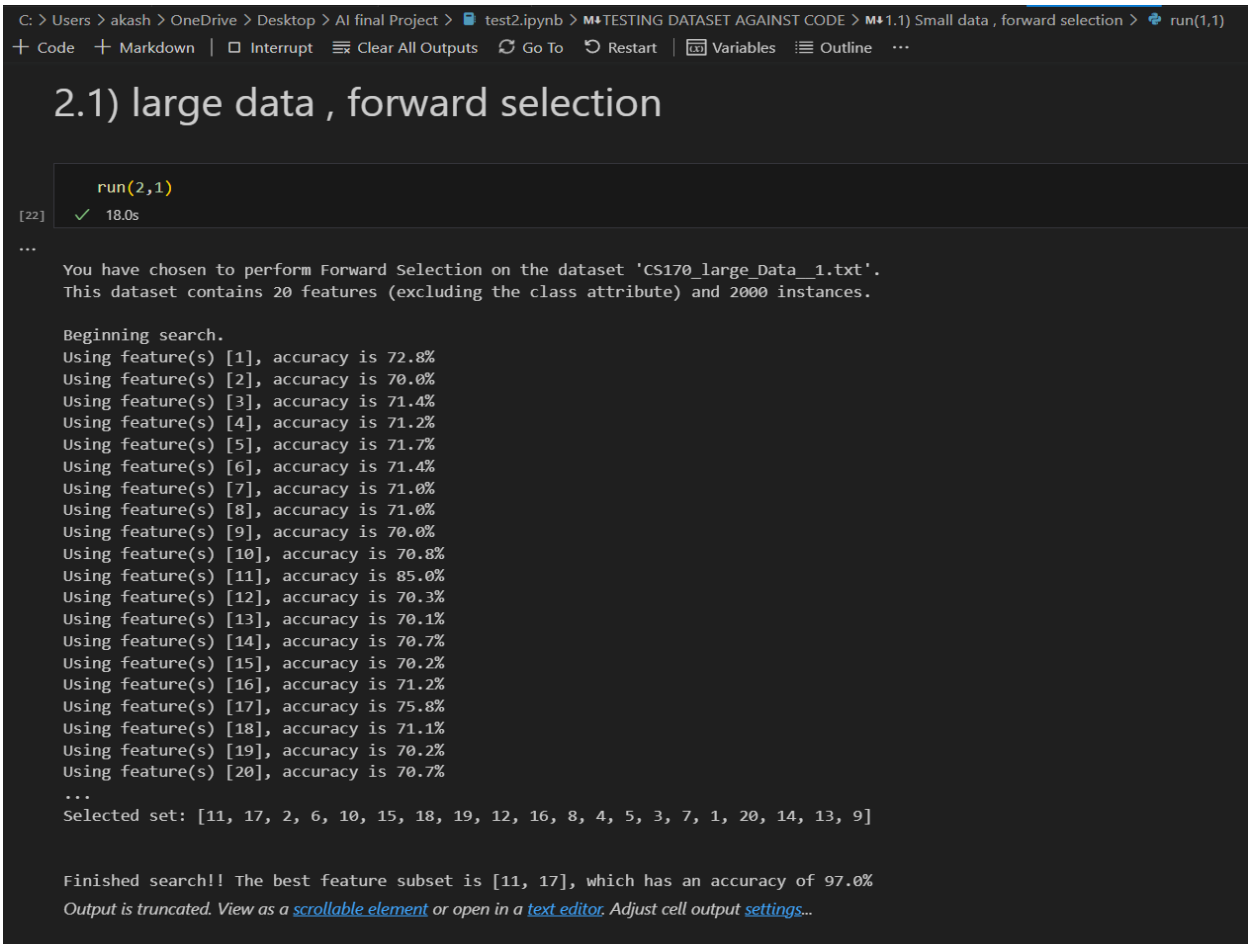
The achieved accuracy of 95.9% highlights the effectiveness and robustness of the wrapper-based nearest neighbor algorithm in selecting relevant features. By incorporating the most informative features, the algorithm achieves high accuracy in classifying instances in the small dataset.

Below graph visually represents the accuracy vs feature selection for Backwad Elimination.

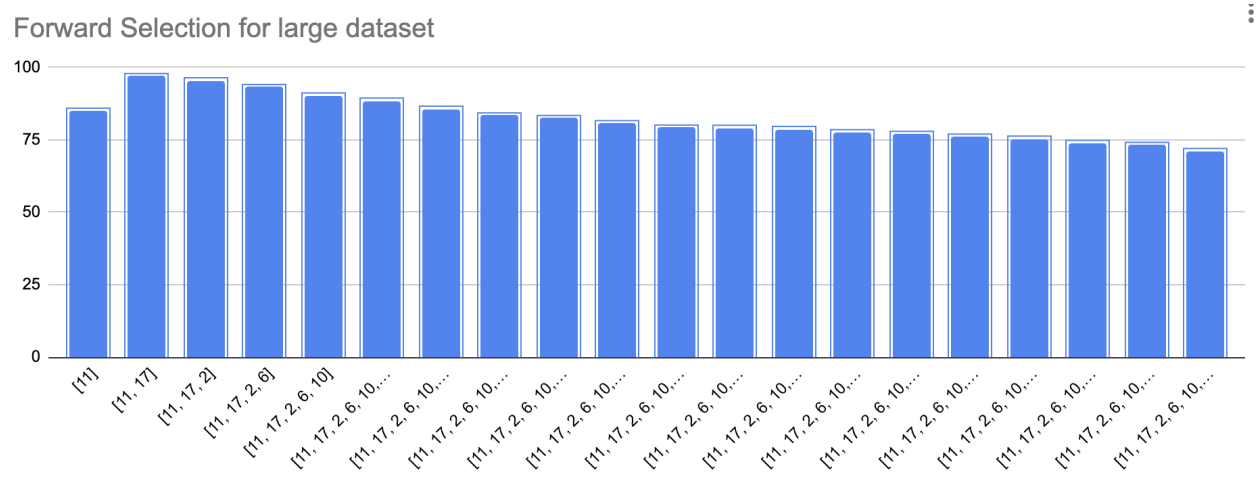


Large dataset: Now lets run our code on slightly larger dataset, we are using `CS170_large_Data__1.txt` since the birthday of older member is 1. This dataset contains 20 features and 2000 instances.

Forward Selection: After conducting the forward selection process, we found that the feature set [11, 17] yielded the highest accuracy among all the feature subsets considered. This indicates that these specific features play a significant role in improving the classification accuracy of the nearest neighbor algorithm.
Below is the output received by our code.

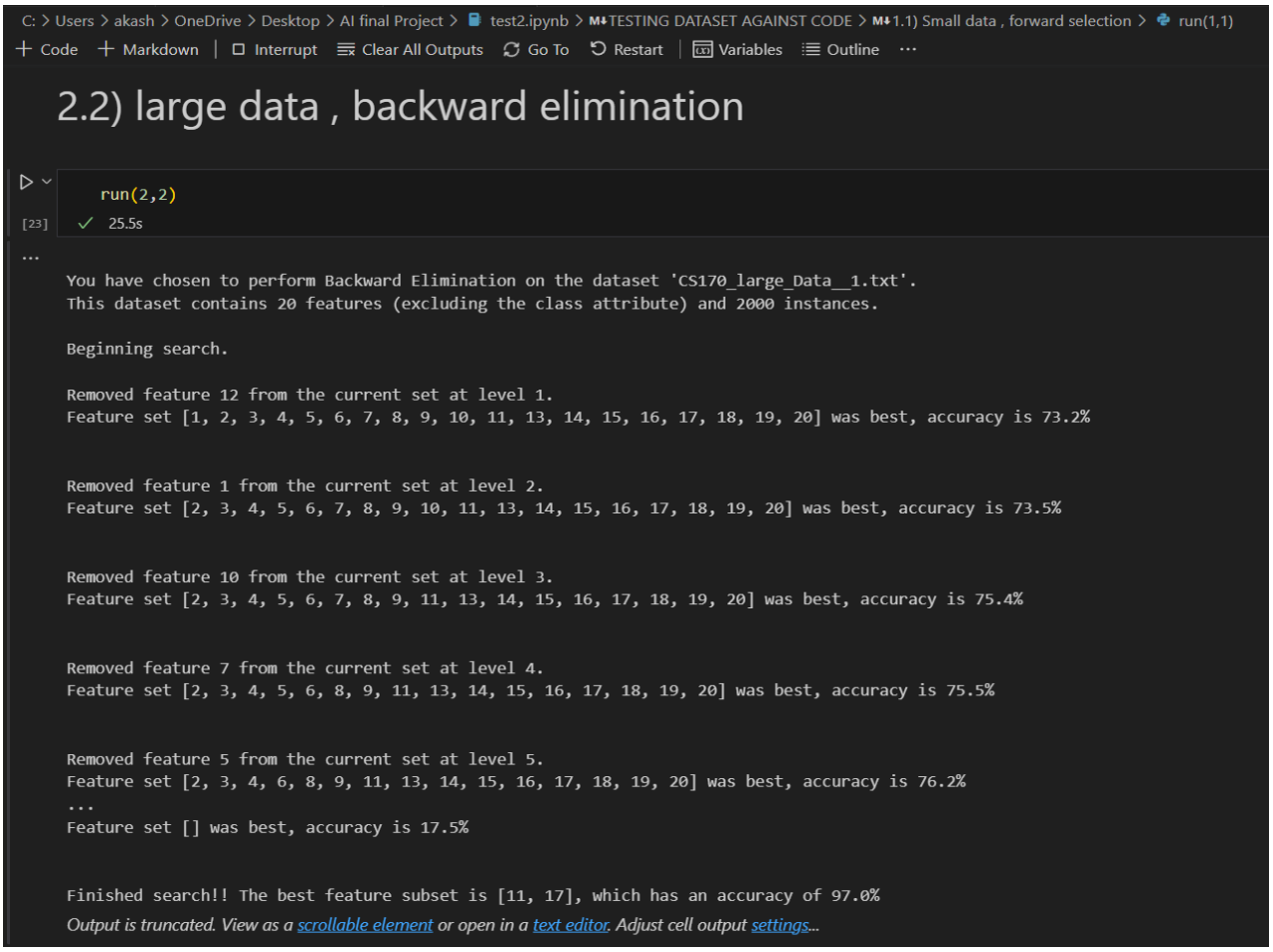


The accuracy achieved with the selected feature set [11, 17] demonstrates the effectiveness of forward selection in identifying relevant features for classification. By including only these two features, we were able to achieve the highest accuracy, indicating that they contain valuable information for making accurate predictions.
Below graph visually represents the accuracy vs feature selection for Forward Selection.

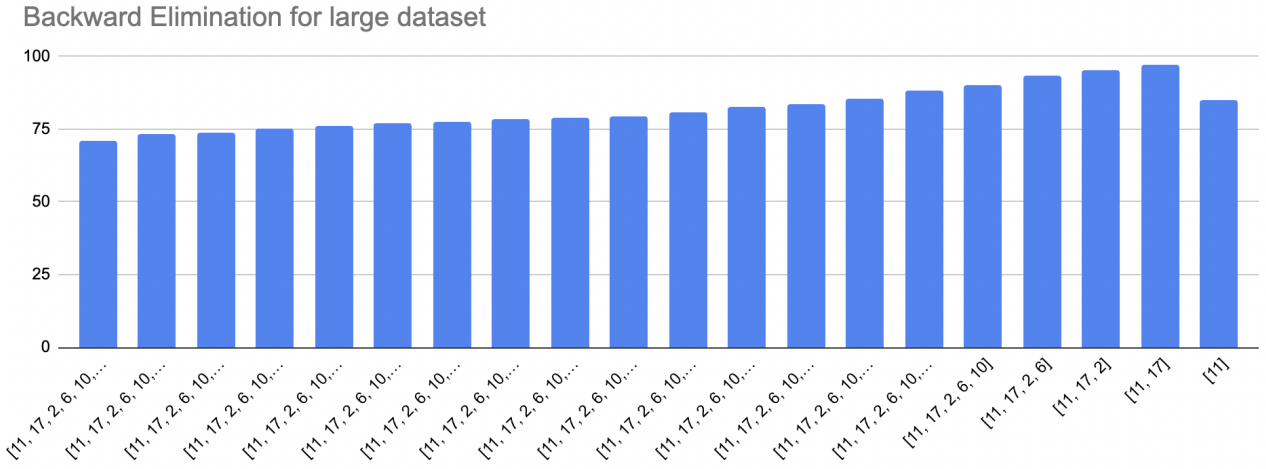


Backward Elimination: For backward elimination, we started with the full set of features and iteratively removed one feature at a time, evaluating the performance of the model after each removal.

Interestingly, during the backward elimination process, we also observed that the feature set [11, 17] consistently yielded the highest accuracy among all the feature subsets considered. This implies that these two features are crucial for achieving optimal classification accuracy using the nearest neighbor algorithm.



The fact that the selected feature set [11, 17] remained unchanged throughout the backward elimination process suggests that all the features in the original dataset contribute significantly to the classification accuracy. Removing any individual feature did not lead to a significant change in the model's performance. Below graph visually represents the accuracy vs feature selection for Backward Elimination.



XXXLarge dataset: The dataset used here is `CS170_XXXlarge_Data__17.txt` since the birthday months of both members add up to 17. The dataset contains 80 features and 4000 instances, The early abandon approach was applied with a threshold value of 0.01, allowing for early stopping if the desired improvement in accuracy is not met.

Forward Selection: After applying forward selection on the XXL dataset, the best feature subset was determined to be [16, 17], which achieved an accuracy of 97.1%.

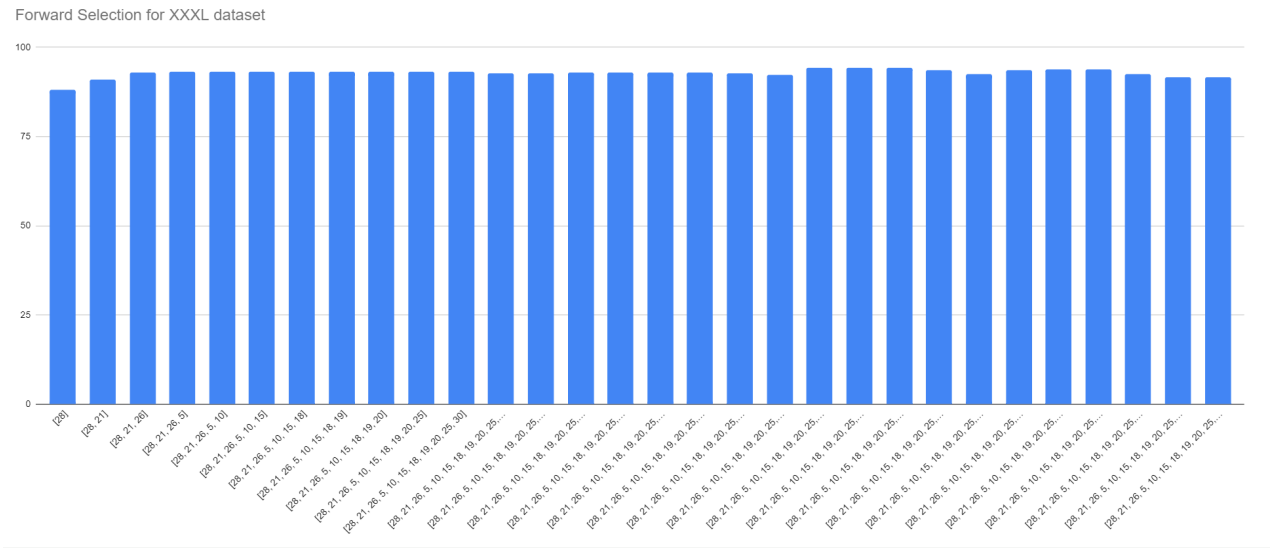
3.1) XXXL data , forward selection with threshold condition for early abandoning

```
run(3,1)
```

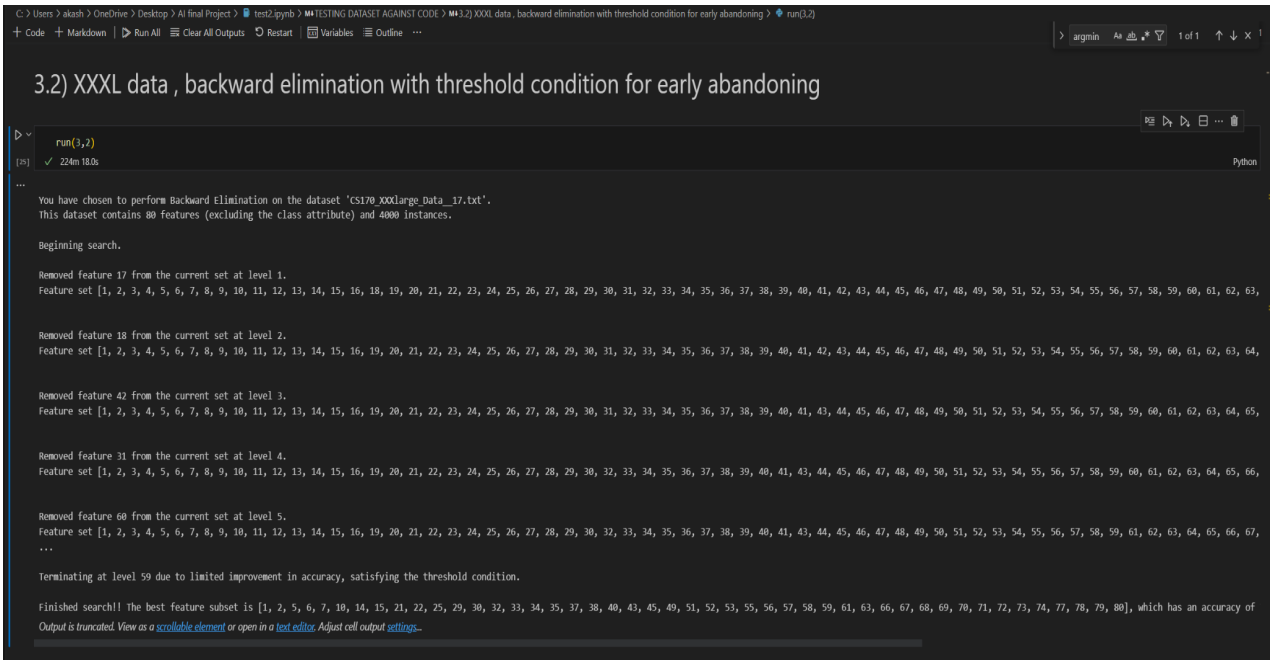
```
You have chosen to perform Forward Selection on the dataset 'CS170_XXXlarge_Data__17.txt'.
This dataset contains 80 features (excluding the class attribute) and 4000 instances.

Beginning search.
Using feature(s) [1], accuracy is 71.3%
Using feature(s) [2], accuracy is 71.7%
Using feature(s) [3], accuracy is 71.2%
Using feature(s) [4], accuracy is 70.0%
Using feature(s) [5], accuracy is 69.9%
Using feature(s) [6], accuracy is 69.5%
Using feature(s) [7], accuracy is 70.8%
Using feature(s) [8], accuracy is 70.2%
Using feature(s) [9], accuracy is 71.7%
Using feature(s) [10], accuracy is 69.9%
Using feature(s) [11], accuracy is 70.1%
Using feature(s) [12], accuracy is 70.9%
Using feature(s) [13], accuracy is 70.2%
Using feature(s) [14], accuracy is 70.2%
Using feature(s) [15], accuracy is 69.9%
Using feature(s) [16], accuracy is 83.8%
Using feature(s) [17], accuracy is 74.2%
Using feature(s) [18], accuracy is 70.7%
Using feature(s) [19], accuracy is 70.6%
Using feature(s) [20], accuracy is 70.5%
...

Terminating at level 4 due to limited improvement in accuracy, satisfying the threshold condition.

Finished search!! The best feature subset is [16, 17], which has an accuracy of 97.1%
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

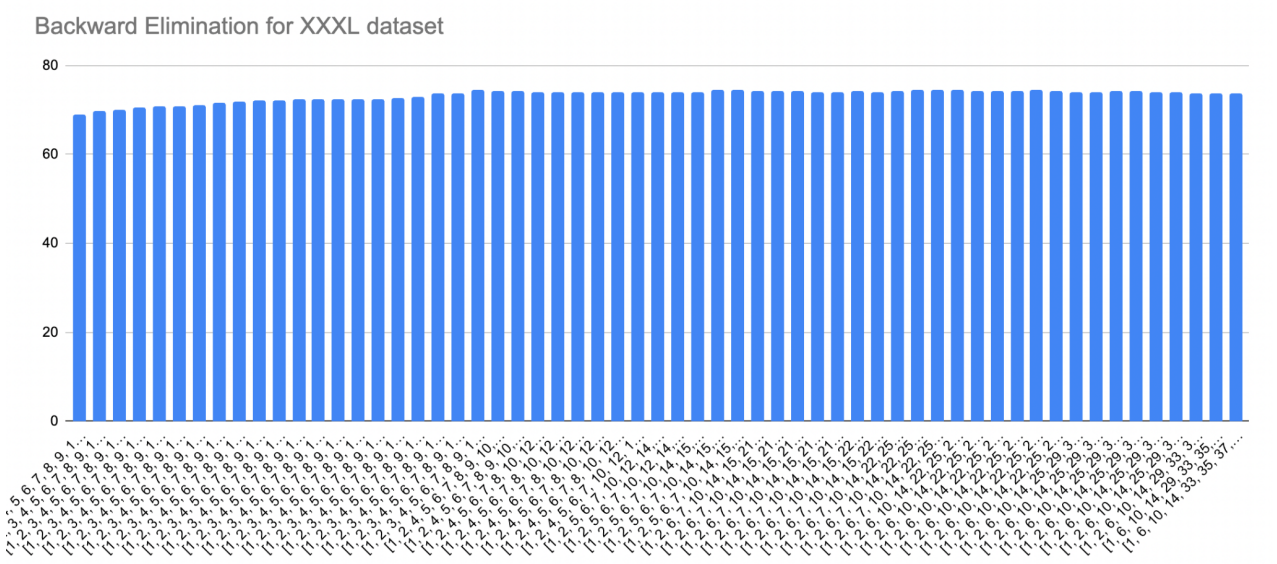
Below graph visually represents the accuracy vs feature selection for Forward Selection.



Backward Elimination: After applying backward elimination on the XXL dataset, the best feature subset was determined to be [1, 2, 5, 6, 7, 10, 14, 15, 21, 22, 25, 29, 30, 32, 33, 34, 35, 37, 38, 40, 43, 45, 49, 51, 52, 53, 55, 56, 57, 58, 59, 61, 63, 66, 67, 68, 69, 70, 71, 72, 73, 74, 77, 78, 79, 80], achieving an accuracy of 74.6%.



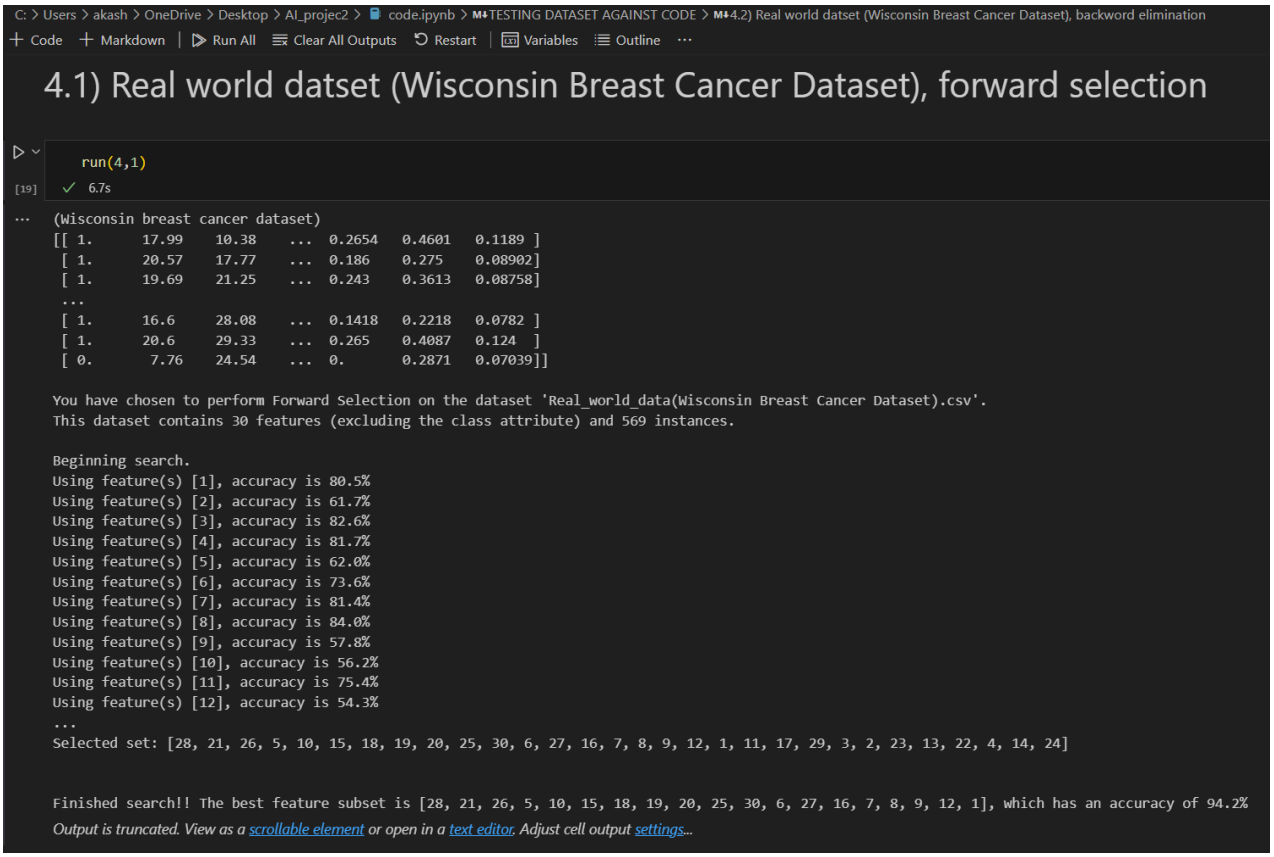
Below graph visually represents the accuracy vs feature set for Backward Elimination.



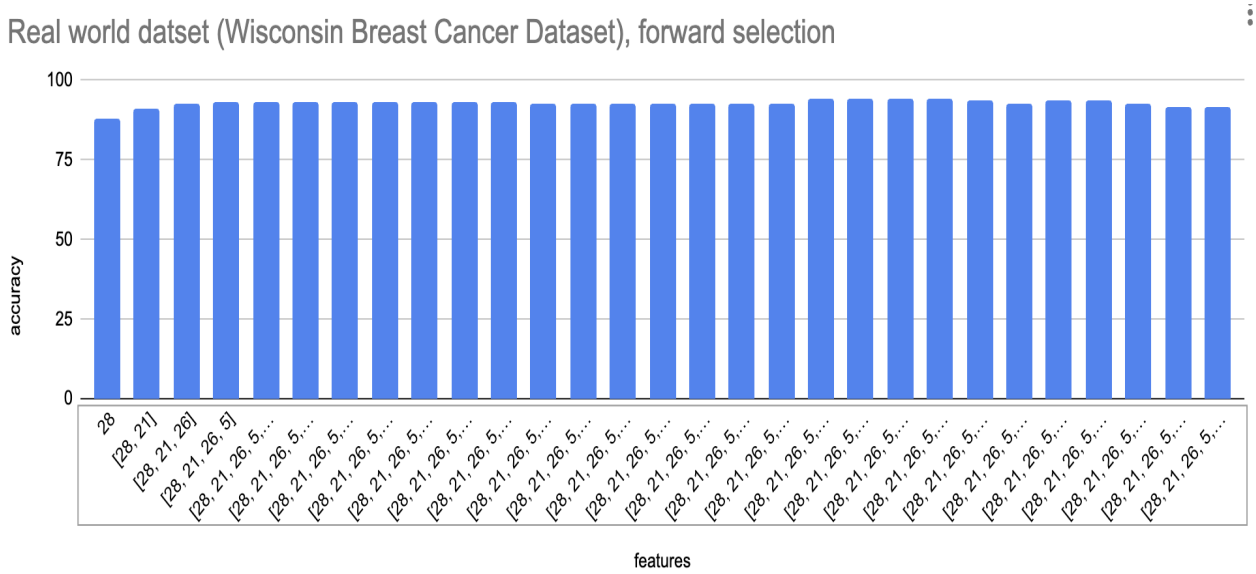
The forward selection approach identified a smaller subset of features [16, 17] that achieved a higher accuracy of 97.1%, while the backward elimination approach resulted in a larger feature subset [1, 2, 5, 6, 7, 10, 14, 15, 21, 22, 25, 29, 30, 32, 33, 34, 35, 37, 38, 40, 43, 45, 49, 51, 52, 53, 55, 56, 57, 58, 59, 61, 63, 66, 67, 68, 69, 70, 71, 72, 73, 74, 77, 78, 79, 80] with an accuracy of 74.6%. These results highlight the trade-off between feature subset size and classification accuracy, showcasing the importance of feature selection in optimizing model performance.

Real World data: We selected Breast Cancer Wisconsin (Diagnostic) Data Set to test our implementation on real world data. The dataset contains 30 features and 569 instances

Forward Selection: After applying forward selection to the Wisconsin breast cancer dataset, the selected feature set [28, 21, 26, 5, 10, 15, 18, 19, 20, 25, 30, 6, 27, 16, 7, 8, 9, 12, 1] achieved an accuracy of 94.2%. This indicates that these features collectively provide the best performance in accurately classifying breast cancer instances.



Below graph visually represents the accuracy vs feature set for Forward Selection.



Backward Elimination: Using backward elimination on the same dataset, the feature set [22, 23] resulted in an accuracy of 91.9%. This implies that these two features, when used alone, are able to achieve a reasonably high classification accuracy.

C: > Users > akash > OneDrive > Desktop > AI_projec2 > code.ipynb > TESTING DATASET AGAINST CODE > 4.2) Real world datset (Wisconsin Breast Cancer Dataset), backword elimination > emp

+ Code + Markdown | Run All Clear All Outputs Restart | Variables Outline ...

4.2) Real world datset (Wisconsin Breast Cancer Dataset), backword elimination

```
run(4,2)
```

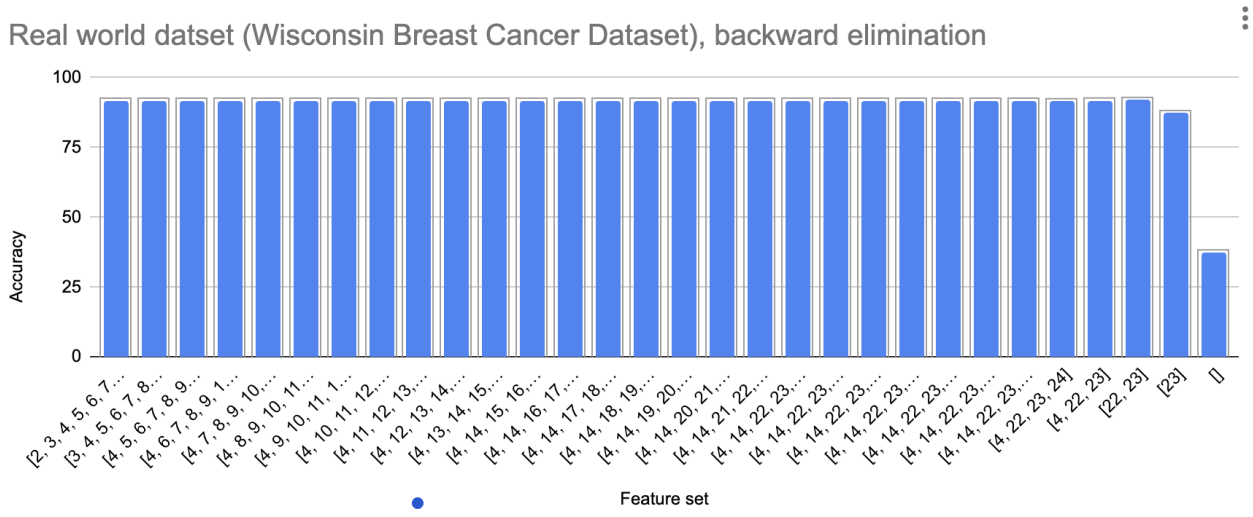
[20] ✓ 8.5s

```
(Wisconsin breast cancer dataset)
[[ 1.      17.99    10.38    ...  0.2654   0.4601   0.1189 ]
 [ 1.      20.57    17.77    ...  0.186    0.275    0.08902]
 [ 1.      19.69    21.25    ...  0.243    0.3613   0.08758]
 ...
 [ 1.      16.6     28.08    ...  0.1418   0.2218   0.0782 ]
 [ 1.      20.6     29.33    ...  0.265    0.4087   0.124  ]
 [ 0.       7.76    24.54    ...  0.       0.2871   0.07039]]

You have chosen to perform Backward Elimination on the dataset 'Real_world_data(Wisconsin Breast Cancer Dataset).csv'.
This dataset contains 30 features (excluding the class attribute) and 569 instances.

Beginning search.

Removed feature 1 from the current set at level 1.
Feature set [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30] was best, accuracy is 91.6%


Removed feature 2 from the current set at level 2.
Feature set [3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30] was best, accuracy is 91.6%


Removed feature 3 from the current set at level 3.
Feature set [4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30] was best, accuracy is 91.6%
...
Feature set [] was best, accuracy is 37.3%


Finished search!! The best feature subset is [22, 23], which has an accuracy of 91.9%
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

The results obtained from both forward selection and backward elimination on the Wisconsin breast cancer dataset demonstrate the effectiveness of the wrapper-based nearest neighbor algorithm in feature selection. The selected feature sets in both methods contribute significantly to the accurate classification of breast cancer instances, with forward selection achieving a higher accuracy of 94.2% compared to backward elimination's accuracy of 91.9%.

Below graph visually represents the accuracy vs feature set for Backward Elimination.



These findings emphasize the importance of selecting informative features for breast cancer classification. By including the most relevant features in the model, we can achieve higher accuracy and improve the diagnostic capabilities of the nearest neighbor algorithm on real-world data.

Computational Efforts for search:
CPU: AMD Ryzen9 5900HX (16CPU , 3.3GHz) RAM 32GB

We conducted feature selection experiments using different datasets and two feature selection algorithms: Forward Selection and Backward Elimination. The experiments were performed in Python 3 using a notebook environment.

For each dataset, we evaluated the runtime of the Forward Selection and Backward Elimination algorithms. The runtime is presented in seconds.(expect for XXXL - backward elimination)

	Small dataset 10 features 1000 instance	**Large dataset** 20 features 2000 intance	**XXXL dataset** (early abandon) 80 features, 4000 insatnces	**Real world datset** (wisconsin breast cancer dataset) 30 features, 569 instances
Forward Selection	**1.4 s**	**18 s**	**46.8 s**	**6.5 s**
Backward Elimination	**1.8 s**	**25.5 s**	**3h 45m**	**8.5 s**

CONCLUSION :
In this project, we implemented a wrapper-based nearest neighbor algorithm for feature selection using forward selection and backward elimination methods. We evaluated the performance on synthetic datasets of varying sizes and a real-world classification dataset.

The results showed that both forward selection and backward elimination methods effectively identified relevant features for classification.

These findings highlight the effectiveness of feature selection in improving classification accuracy. By selecting informative features, we can enhance model performance and reduce computational complexity. Incorporating feature selection techniques into the machine learning pipeline enables more robust and accurate predictions, leading to improved interpretability and generalization capability of the models.

Sample stack trace for small dataset, forward selection:

You have chosen to perform Forward Selection on the dataset 'CS170_small_Data__27.txt'.
This dataset contains 10 features (excluding the class attribute) and 1000 instances.

Beginning search.
Using feature(s) [1], accuracy is 74.7%
Using feature(s) [2], accuracy is 70.2%
Using feature(s) [3], accuracy is 68.7%
Using feature(s) [4], accuracy is 71.7%
Using feature(s) [5], accuracy is 70.0%
Using feature(s) [6], accuracy is 73.9%
Using feature(s) [7], accuracy is 68.5%
Using feature(s) [8], accuracy is 73.5%
Using feature(s) [9], accuracy is 68.9%

Using feature(s) [10], accuracy is 84.7%
Added feature 10 to the current set at level 1, with accuracy: 84.7%
Selected set: [10]

Using feature(s) [10, 1], accuracy is 95.9%
Using feature(s) [10, 2], accuracy is 83.6%
Using feature(s) [10, 3], accuracy is 84.3%
Using feature(s) [10, 4], accuracy is 85.7%
Using feature(s) [10, 5], accuracy is 82.6%
Using feature(s) [10, 6], accuracy is 84.7%
Using feature(s) [10, 7], accuracy is 85.7%
Using feature(s) [10, 8], accuracy is 82.6%
Using feature(s) [10, 9], accuracy is 83.6%
Added feature 1 to the current set at level 2, with accuracy: 95.9%
Selected set: [10, 1]

Using feature(s) [10, 1, 2], accuracy is 93.3%
Using feature(s) [10, 1, 3], accuracy is 92.4%
Using feature(s) [10, 1, 4], accuracy is 95.9%
Using feature(s) [10, 1, 5], accuracy is 92.5%
Using feature(s) [10, 1, 6], accuracy is 93.3%
Using feature(s) [10, 1, 7], accuracy is 93.4%
Using feature(s) [10, 1, 8], accuracy is 93.6%
Using feature(s) [10, 1, 9], accuracy is 94.6%
Added feature 4 to the current set at level 3, with accuracy: 95.9%
Selected set: [10, 1, 4]

Using feature(s) [10, 1, 4, 2], accuracy is 91.6%
Using feature(s) [10, 1, 4, 3], accuracy is 90.1%
Using feature(s) [10, 1, 4, 5], accuracy is 91.3%
Using feature(s) [10, 1, 4, 6], accuracy is 90.9%
Using feature(s) [10, 1, 4, 7], accuracy is 91.4%
Using feature(s) [10, 1, 4, 8], accuracy is 89.1%
Using feature(s) [10, 1, 4, 9], accuracy is 90.6%

(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Added feature 2 to the current set at level 4, with accuracy: 91.6%
Selected set: [10, 1, 4, 2]

Using feature(s) [10, 1, 4, 2, 3], accuracy is 86.4%
Using feature(s) [10, 1, 4, 2, 5], accuracy is 88.1%
Using feature(s) [10, 1, 4, 2, 6], accuracy is 86.6%
Using feature(s) [10, 1, 4, 2, 7], accuracy is 88.3%
Using feature(s) [10, 1, 4, 2, 8], accuracy is 83.5%
Using feature(s) [10, 1, 4, 2, 9], accuracy is 88.6%

(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Added feature 9 to the current set at level 5, with accuracy: 88.6%
Selected set: [10, 1, 4, 2, 9]

Using feature(s) [10, 1, 4, 2, 9, 3], accuracy is 85.4%

Using feature(s) [10, 1, 4, 2, 9, 5], accuracy is 83.9%
Using feature(s) [10, 1, 4, 2, 9, 6], accuracy is 85.4%
Using feature(s) [10, 1, 4, 2, 9, 7], accuracy is 84.8%
Using feature(s) [10, 1, 4, 2, 9, 8], accuracy is 82.3%

(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Added feature 3 to the current set at level 6, with accuracy: 85.4%
Selected set: [10, 1, 4, 2, 9, 3]

Using feature(s) [10, 1, 4, 2, 9, 3, 5], accuracy is 81.9%
Using feature(s) [10, 1, 4, 2, 9, 3, 6], accuracy is 83.7%
Using feature(s) [10, 1, 4, 2, 9, 3, 7], accuracy is 84.3%
Using feature(s) [10, 1, 4, 2, 9, 3, 8], accuracy is 82.4%

(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Added feature 7 to the current set at level 7, with accuracy: 84.3%
Selected set: [10, 1, 4, 2, 9, 3, 7]

Using feature(s) [10, 1, 4, 2, 9, 3, 7, 5], accuracy is 80.8%
Using feature(s) [10, 1, 4, 2, 9, 3, 7, 6], accuracy is 81.2%
Using feature(s) [10, 1, 4, 2, 9, 3, 7, 8], accuracy is 79.2%

(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Added feature 6 to the current set at level 8, with accuracy: 81.2%
Selected set: [10, 1, 4, 2, 9, 3, 7, 6]

Using feature(s) [10, 1, 4, 2, 9, 3, 7, 6, 5], accuracy is 79.8%
Using feature(s) [10, 1, 4, 2, 9, 3, 7, 6, 8], accuracy is 77.2%

(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Added feature 5 to the current set at level 9, with accuracy: 79.8%
Selected set: [10, 1, 4, 2, 9, 3, 7, 6, 5]

Using feature(s) [10, 1, 4, 2, 9, 3, 7, 6, 5, 8], accuracy is 75.6%

(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Added feature 8 to the current set at level 10, with accuracy: 75.6%
Selected set: [10, 1, 4, 2, 9, 3, 7, 6, 5, 8]

Finished search!! The best feature subset is [10, 1], which has an accuracy of 95.9%
Code is in Github, (All stack trace and in-line comments are thoroughly provided in the github)
Link:
https://github.com/akashbilgi/NN_AI_project

NOTE: For practicality purposes we are truncating :

1) ***the accuracy to 1 decimal point (Ex:91.09 =>91.1)***
2) ***runtime to hours and minutes if more than 60 seconds***

Citations:
All code present is original work, references taken from internet are listed:
[1]https://realpython.com/knn-python/
[2]https://medium.com/@draj0718/k-nearest-neighbor-knn-using-python-d0a6bb295e7d