Project 2: Feature Selection with Nearest Neighbor

Student Name: Steven Nguyen SID: 862114346

Solution: <the datasets are your uniquely assigned datasets>

Dataset	Best Feature Set	Accuracy
Small Number: 146	Forward Selection = {2, 4, 1, 10, 7}	0.98
	Backward Elimination = {3, 6, 8, 1}	0.85
	Custom Algorithm = Not implemented	n/a
Large Number: 146	Forward Selection = {17, 39, 20, 14, 36, 26}	0.82
	Backward Elimination = {6, 25, 16, 37}	0.87
	Custom Algorithm = Not implemented	n/a

-------------------------------<Begin Report>-----------------------------------

In completing this project, I consulted following resources:

https://www.geeksforgeeks.org/program-for-conversion-of-32-bits-single-precision-ieee-754-floating-point-representation/

https://quantifyinghealth.com/stepwise-selection/

https://towardsai.net/p/data-science/how-when-and-why-should-you-normalize-standardize-rescale-your-data-3f083def38ff

I. Challenges

- One challenge I faced during this project was completely understanding the algorithm before I could correctly implement it in code.
- The debugging process while trying to get the program to run was also hard because I would always run into technical issues during the coding process.
- Allocating time to finish this project was also hard as many of my other classes had assignments due during this time period.
- I was never able to fix the bug where I kept getting the same accuracy for each feature. I believe the problem was that I kept looping too much and compared each feature with everything else at each level.
- I believe my code for phase 2 was correct (loading data, getting accuracy, and modifying columns) but after combining everything together in phase 3 in the forward_selection/back_selection functions things didn't work as intended.

II. Code Design

- searchAlgorithms.cpp
    - void forward_selection() - forward selection algorithm
    - void backward_selection() - backward selection algorithm
    - int random_num() - phase 1 function that returned a random accuracy
    - void loadData() - loads data from the txt files
    - double getAccuracy() - gets accuracy of feature using leave one out cross-validation
    - void modifyData() - makes columns of data 0 if feature is not selected
    - double calculate_distance - finds euclidian distance of features

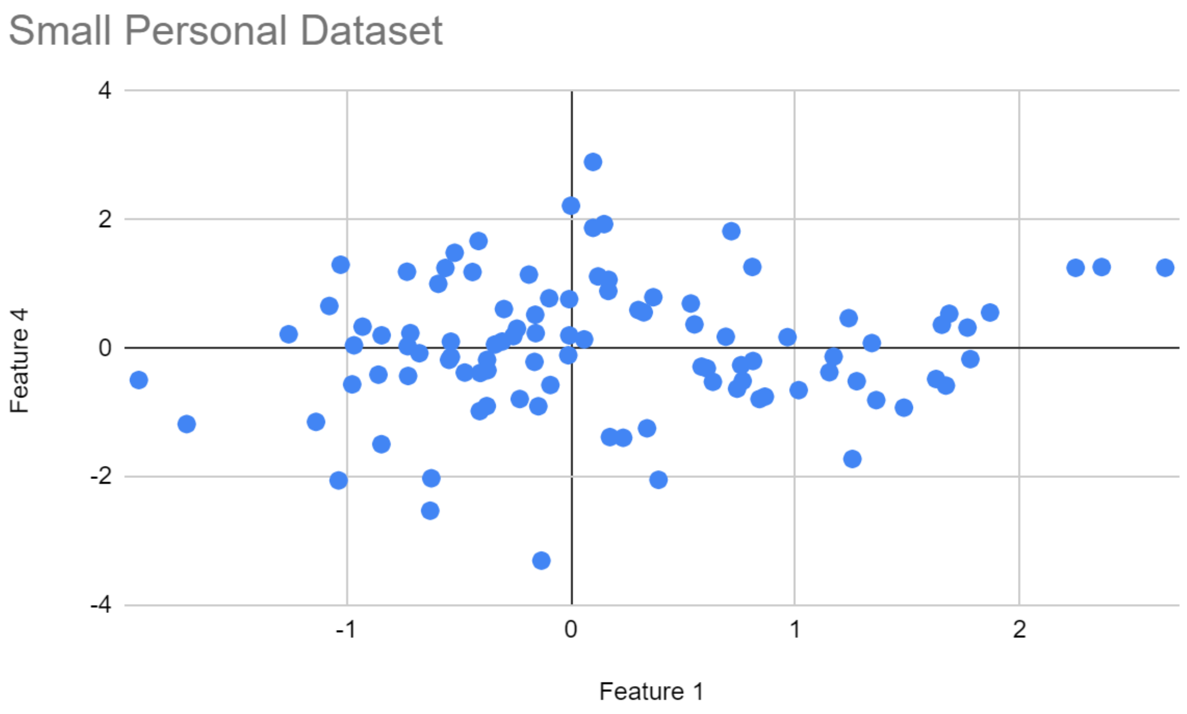
III. Dataset details

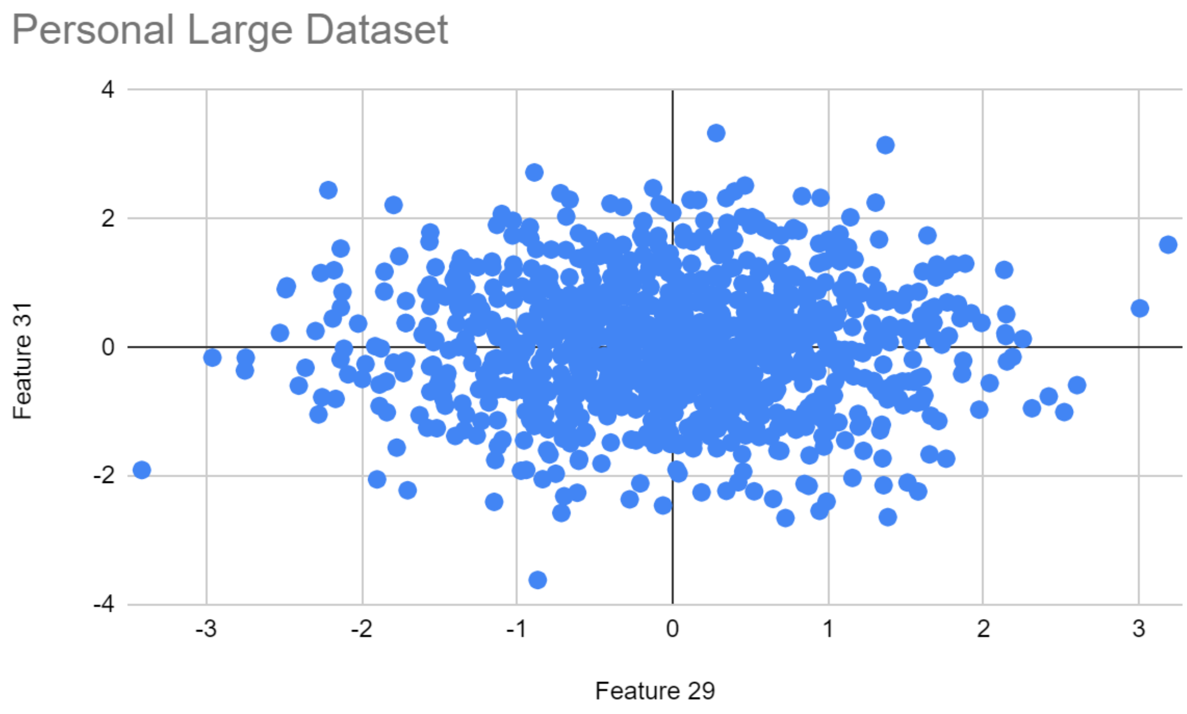
Your Small Dataset: Number of features, number of instances

- 10 features, 100 instances

Your Large Dataset: Number of features, number of instances

- 40 features, 1000 instances





IV. Algorithms

1. Forward Selection - Starts with an empty set and continues to add features one by one into the model. At each level, the algorithm chooses the feature that contributes the highest accuracy (improvment) towards the model. The algorithm finishes after all the features have been added in the final set. It is then able to find the feature subset that contributed the highest total accuracy.
2. Backward Elimination - The opposite of forward selection. Starts with the final set of features and continues to remove features one by one. At each level, the algorithm removes the feature that contributes the highest accuracy (improvement) towards the model. The algorithm finishes after all the features have been removed and are left with the empty set. It is then able to find the feature subset that contributed the highest total accuracy.

V. Analysis

Experiment 1: Comparing Forward Selection vs Backward Elimination.

Pros Forward Selection: Works better when there are many features to select from. This is because forward selection starts with the empty set, allowing it to continue to add the most important features and considering the full model later.

Pros Backward Elimination: Works better when considering features that are correlated. This is because starting with the completed set allows you to take into account all features. Backward elimination will keep these correlated features in the model where forward selection might miss them.

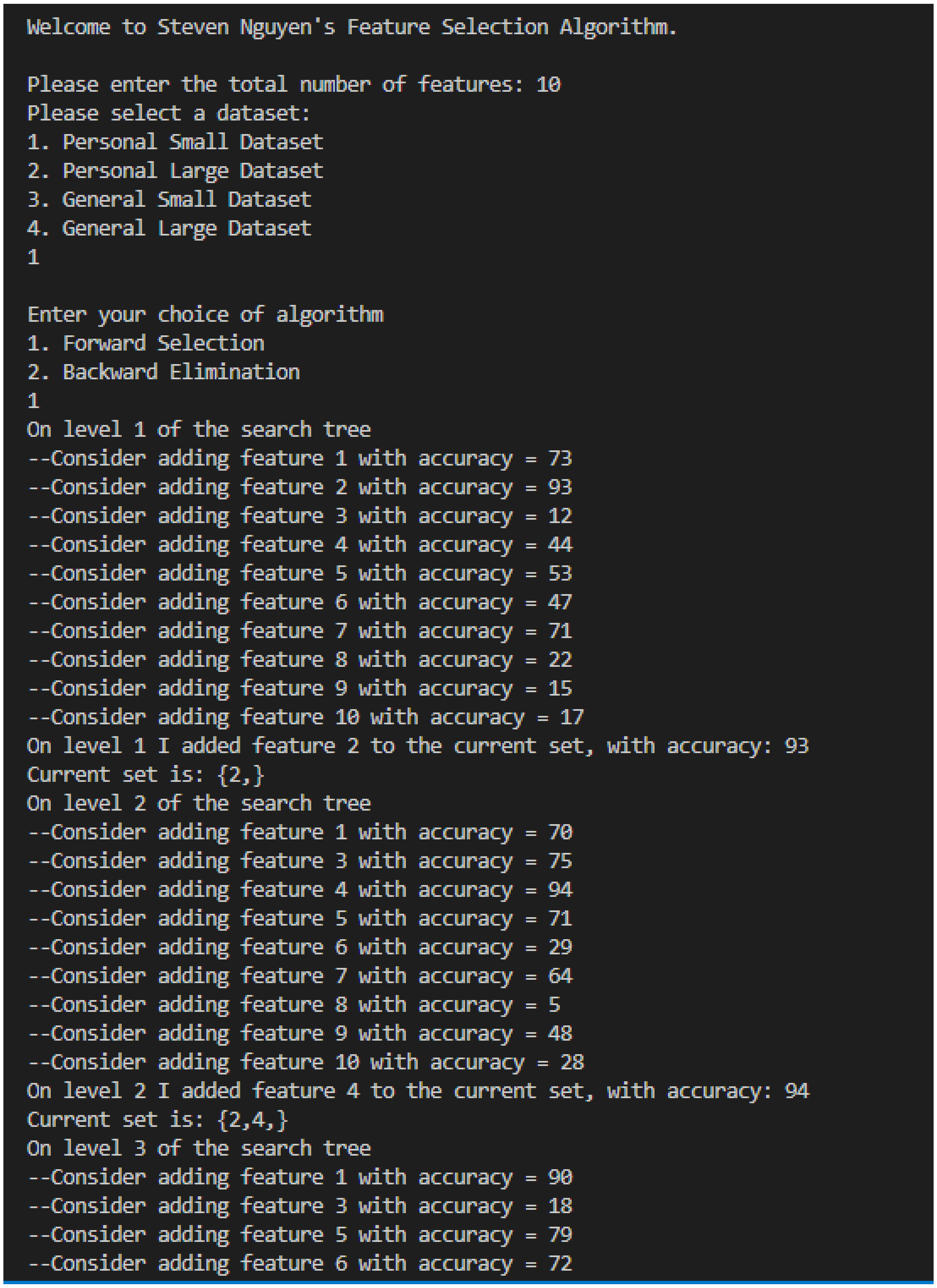
Experiment 2: Effect of normalization

Depending on the dataset, normalizing your data could help your accuracy change. A feature with a lot more instances in an unnormalized dataset would have a higher accuracy score than its counterparts due to its size. However, after normalizing the data, each feature will operate on the same range of 0-1 where it will show if the larger feature truly is more accurate not just due to size.

Experiment 3: Effect of number neighbors (k)

By increasing k nearest neighbors the accuracy of each feature will definitely increase. This is because comparing a point to more points around it will give you a better idea of what to classify it as. Say the k nearest neighbor was 1 and the nearest neighbor to a given object was of class blue. Say the next 3 nearest neighbors were of class green, using 4 as the k nearest neighbor instead of 1 would have classified our point as green instead which is more accurate than blue.

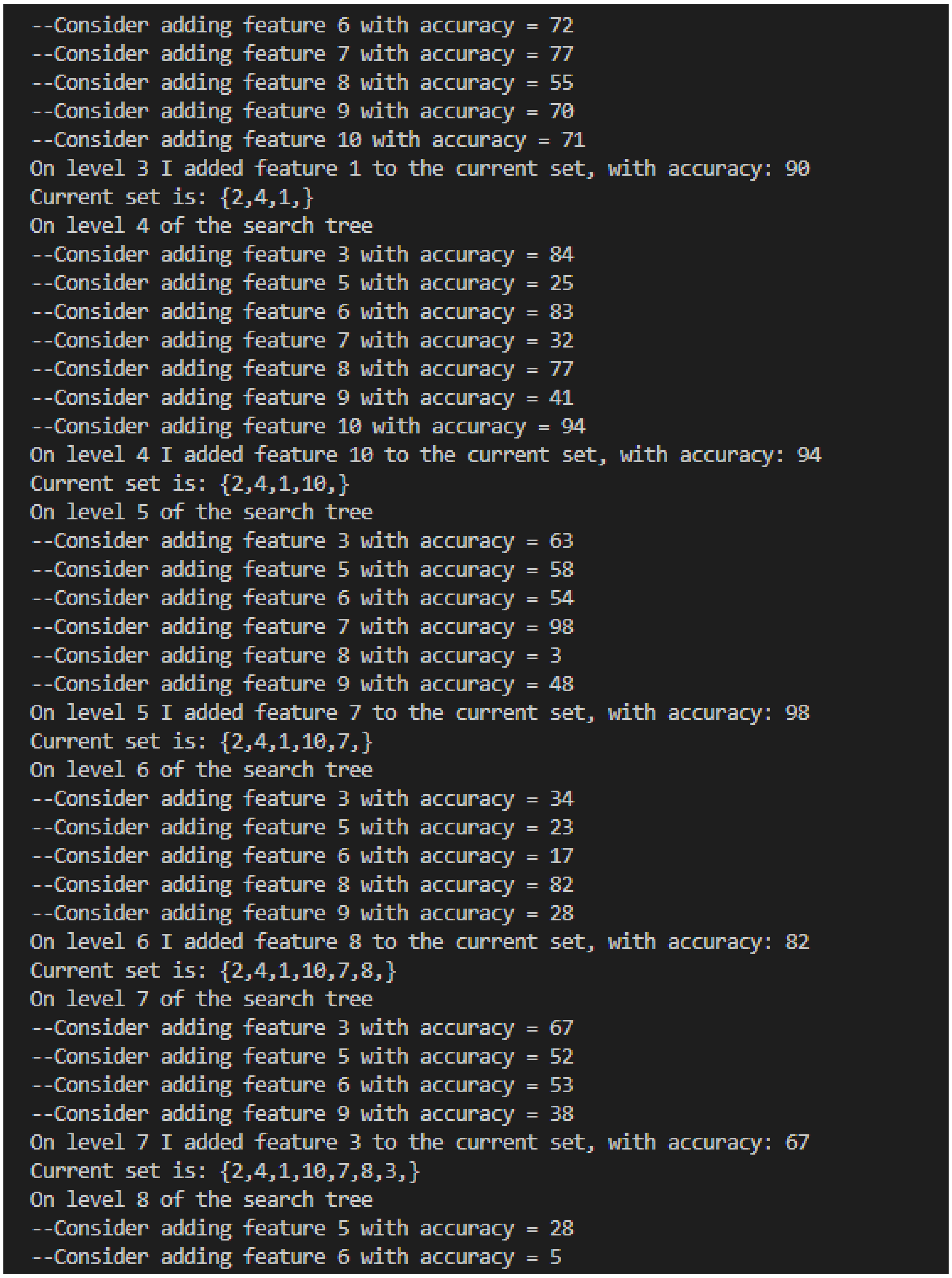
VI. Trace of your small dataset



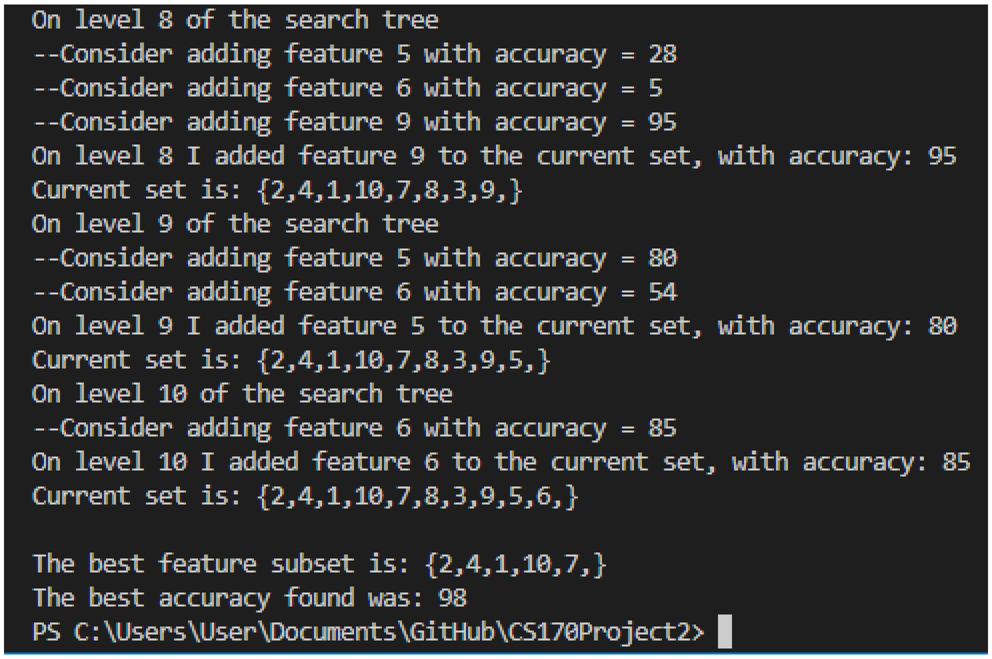
```
Welcome to Steven Nguyen's Feature Selection Algorithm.

Please enter the total number of features: 10
Please select a dataset:
1. Personal Small Dataset
2. Personal Large Dataset
3. General Small Dataset
4. General Large Dataset
1

Enter your choice of algorithm
1. Forward Selection
2. Backward Elimination
1
On level 1 of the search tree
--Consider adding feature 1 with accuracy = 73
--Consider adding feature 2 with accuracy = 93
--Consider adding feature 3 with accuracy = 12
--Consider adding feature 4 with accuracy = 44
--Consider adding feature 5 with accuracy = 53
--Consider adding feature 6 with accuracy = 47
--Consider adding feature 7 with accuracy = 71
--Consider adding feature 8 with accuracy = 22
--Consider adding feature 9 with accuracy = 15
--Consider adding feature 10 with accuracy = 17
On level 1 I added feature 2 to the current set, with accuracy: 93
Current set is: {2,}
On level 2 of the search tree
--Consider adding feature 1 with accuracy = 70
--Consider adding feature 3 with accuracy = 75
--Consider adding feature 4 with accuracy = 94
--Consider adding feature 5 with accuracy = 71
--Consider adding feature 6 with accuracy = 29
--Consider adding feature 7 with accuracy = 64
--Consider adding feature 8 with accuracy = 5
--Consider adding feature 9 with accuracy = 48
--Consider adding feature 10 with accuracy = 28
On level 2 I added feature 4 to the current set, with accuracy: 94
Current set is: {2,4,}
On level 3 of the search tree
--Consider adding feature 1 with accuracy = 90
--Consider adding feature 3 with accuracy = 18
--Consider adding feature 5 with accuracy = 79
--Consider adding feature 6 with accuracy = 72
```



```
--Consider adding feature 6 with accuracy = 72
--Consider adding feature 7 with accuracy = 77
--Consider adding feature 8 with accuracy = 55
--Consider adding feature 9 with accuracy = 70
--Consider adding feature 10 with accuracy = 71
On level 3 I added feature 1 to the current set, with accuracy: 90
Current set is: {2,4,1,}
On level 4 of the search tree
--Consider adding feature 3 with accuracy = 84
--Consider adding feature 5 with accuracy = 25
--Consider adding feature 6 with accuracy = 83
--Consider adding feature 7 with accuracy = 32
--Consider adding feature 8 with accuracy = 77
--Consider adding feature 9 with accuracy = 41
--Consider adding feature 10 with accuracy = 94
On level 4 I added feature 10 to the current set, with accuracy: 94
Current set is: {2,4,1,10,}
On level 5 of the search tree
--Consider adding feature 3 with accuracy = 63
--Consider adding feature 5 with accuracy = 58
--Consider adding feature 6 with accuracy = 54
--Consider adding feature 7 with accuracy = 98
--Consider adding feature 8 with accuracy = 3
--Consider adding feature 9 with accuracy = 48
On level 5 I added feature 7 to the current set, with accuracy: 98
Current set is: {2,4,1,10,7,}
On level 6 of the search tree
--Consider adding feature 3 with accuracy = 34
--Consider adding feature 5 with accuracy = 23
--Consider adding feature 6 with accuracy = 17
--Consider adding feature 8 with accuracy = 82
--Consider adding feature 9 with accuracy = 28
On level 6 I added feature 8 to the current set, with accuracy: 82
Current set is: {2,4,1,10,7,8,}
On level 7 of the search tree
--Consider adding feature 3 with accuracy = 67
--Consider adding feature 5 with accuracy = 52
--Consider adding feature 6 with accuracy = 53
--Consider adding feature 9 with accuracy = 38
On level 7 I added feature 3 to the current set, with accuracy: 67
Current set is: {2,4,1,10,7,8,3,}
On level 8 of the search tree
--Consider adding feature 5 with accuracy = 28
--Consider adding feature 6 with accuracy = 5
```



```
On level 8 of the search tree
--Consider adding feature 5 with accuracy = 28
--Consider adding feature 6 with accuracy = 5
--Consider adding feature 9 with accuracy = 95
On level 8 I added feature 9 to the current set, with accuracy: 95
Current set is: {2,4,1,10,7,8,3,9,}
On level 9 of the search tree
--Consider adding feature 5 with accuracy = 80
--Consider adding feature 6 with accuracy = 54
On level 9 I added feature 5 to the current set, with accuracy: 80
Current set is: {2,4,1,10,7,8,3,9,5,}
On level 10 of the search tree
--Consider adding feature 6 with accuracy = 85
On level 10 I added feature 6 to the current set, with accuracy: 85
Current set is: {2,4,1,10,7,8,3,9,5,6,}

The best feature subset is: {2,4,1,10,7,}
The best accuracy found was: 98
PS C:\Users\User\Documents\GitHub\CS170Project2>
```