Project 2: Feature Selection with Nearest Neighbor

Student Name: Kevin Gao SID: 862138776

Solution: <124>

Dataset	Best Feature Set	Accuracy
Small Number: <124>	Forward Selection = {8,2,1}	0.95
	Backward Elimination = {2,3,4,6,7,8,10}	0.85
	Custom Algorithm = Not implemented	N/A
Large Number: <124>	Forward Selection = {8,20}	0.966
	Backward Elimination = {1,3,4,5,6,7,9,10, 11,12,13,14,15,16,18,19,22,23,24,26,28,30,32, 33,34,35,37,38,39,40}	0.77
	Custom Algorithm = Not implemented	N/A

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

In completing this project, I consulted following resources:

https://www.tutorialspoint.com/mean-and-median-of-a-matrix-in-cplusplus

https://www.tutorialspoint.com/cplusplus-program-to-calculate-standard-deviation\

https://www.statology.org/z-score-normalization/

Lecture Slides

Discussion for Project Part 2

Partner: Jordan Sam

Solution: Initial

Dataset	Best Feature Set	Accuracy
Small Number: Initial	Forward Selection = {5,3}	0.92
	Backward Elimination = {2,4,5,7,10}	0.82
	Custom Algorithm = Not implemented	N/A
Large Number: Initial	Forward Selection = {27,1}	0.955
	Backward Elimination = {2,3,4,5,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40},	0.722
	Custom Algorithm = Not implemented	N/A

I. Introduction

This project is a feature selection program that uses Greedy Forward Selection and Backwards Elimination feature selection, Nearest-Neighbor Classification, and Leave-One-Out-Validation to train and test a given dataset with a set class and features. The goal is to select features that will result in the most accurate classification.

II. Challenges

Each stage was difficult to integrate into the existing program, the first stage was not too hard to think through, but the implementation of the validator and the evaluator were especially difficult for me. In particular, transforming the existing instances of data into both testing and training data was hard to figure out. The most difficult challenge for this project was thinking about how everything was supposed to fit together for me. Which classes and files should include which functions, what the overall data flow looked like, and having a lot of functions/variables to code and keep track of was difficult and confusing.

III. Code Design

main.cpp - Holds the initial user menu and takes in user/file input. Decides which feature selection algorithm to use and calls on the Normalizer and Validator class to work the data and train on the data before calling the specified feature selection algorithm.

Node.h - Each Node represents a feature subset to be explored. Nodes contain the feature subset, their own children, feature accuracy scores, and a validator. The Node class also contains an evaluator function to assign scores and a printFeatures function.

Normalizer.h - contains stringToDouble to convert file data from string to double form. Mainly contains a normalize function to z-score normalize the read data.

Classifier.h - Holds a matrix training instances and a vector of training labels. Classifies nodes with feature subsets based on Nearest Neighbor using Euclidean distance.

Validator.h - Holds a matrix of instances and a vector of training labels, as well as a Classifier pointer. This is where leave-one-out-validation happens. The instances in the program are separated into training and testing data, then the data is used to train and is tested.

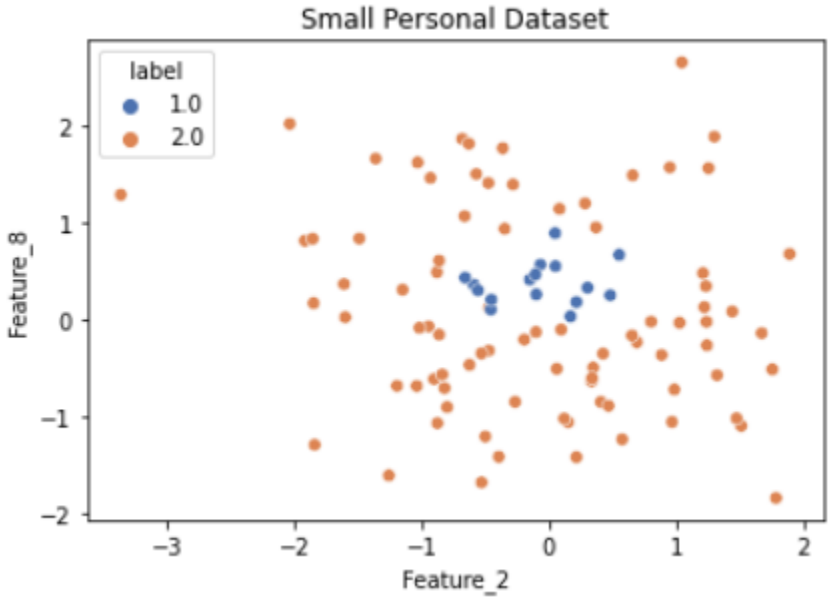
Greedy.h - Holds a starting head node and a validator pointer. Contains both the forward selection algorithm and the backwards elimination algorithm.

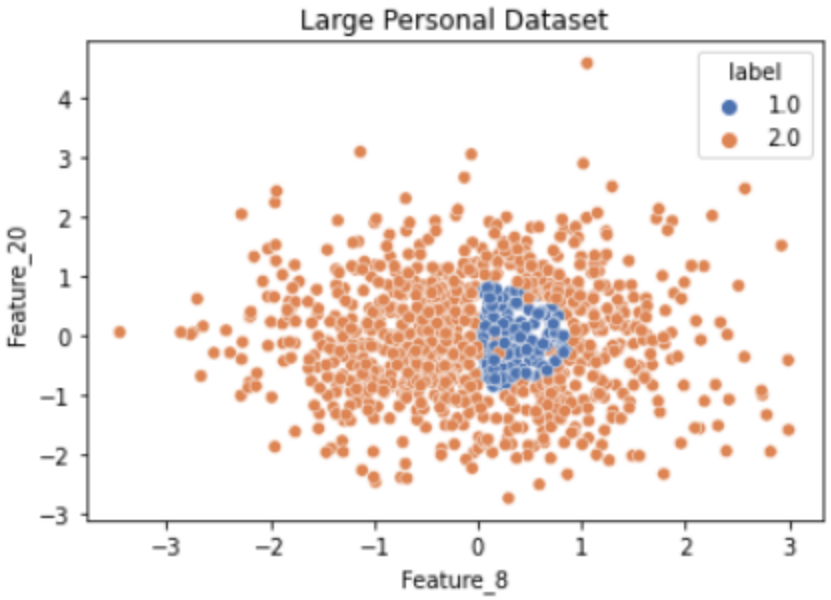
IV. Dataset details

Your Small Dataset (124): 10 features, 100 instances

Your Large Dataset (124): 40 features, 1000 instances

Plot some features and color code them by class and explore your dataset.





V. Algorithms

1. Forward Selection
    a. Given a set of features, evaluate the accuracy of each feature and choose the most accurate one to expand (greedy)
    b. create feature subsets with existing features and the most accurate one, expand the most accurate subset
    c. repeat with larger subsets until the highest accuracy for a feature subset is found (the parent subset is higher than all the kids or the highest accuracy is all the features included in a subset)
2. Backward Elimination
    a. Given a set of features, evaluate the accuracy of all the features in a large feature subset
    b. Remove one feature from the large subset and create smaller subsets of current feature subset size - 1 with one feature removed each.
    c. Repeat with smaller subsets until the highest accuracy for a feature subset is found (the parent subset is higher than all the kids or the highest accuracy is all the features included in a subset)

VI. Analysis

Experiment 1: Comparing Forward Selection vs Backward Elimination.

Compare accuracy with no feature selection vs with feature selection.

In most cases the accuracy with feature selection is higher than the accuracy without feature selection.

With no feature selection accuracy:

	Small Dataset (124)	Large Dataset (124)
Forward Selection	41%	85%
Backward Elimination	78%	67.6%

As shown, when compared to the feature selection accuracy on the first page and the below table, all the accuracies without feature selection are worse.

Compare feature set and accuracy for forward selection vs backward elimination.

With feature selection accuracy:

	Small Dataset (124)	Large Dataset (124)
Forward Selection	95%	96.6%
Backward Elimination	85%	77%

It seems that in this case the forward selection algorithm resulted in higher accuracies compared to backward elimination. This may be because most of the time, instances of data can be categorized into classes based on a few features rather than almost all the features.

Talk about pros and cons of both algorithms.

Both algorithms have their uses, when instances have a lot of features that go into classification, backwards elimination might be better, but when there are simple instances that are easily distinguishable, forward selection is better. Forward selection takes a lot less time while backward elimination takes a lot more time due to multiple features being worked with at the start.

VII. Conclusion

In everyday cases that are simpler, forward selection should be used to select features instead of backward elimination. Forward selection takes a lot less time to run. Potential improvements to be made in feature selection are to use more neighbors for the nearest neighbor algorithm or exploring different ways to validate data besides leave-one-out-validation.

VIII. Trace of your small dataset

```
Welcome to Kevin Gao's Feature Selection Algorithm
Type in the name of the file to test: CS170_Spring_2022_Small_data__124.txt

Type the number of the algorithm you want to run:

1 - Forward Selection
2 - Backward Elimination

1
This dataset has 10 features (not including the class attribute), with 100 instances.
Please wait while I normalize the data...
   Done!

Running nearest neighbor with no features (default rate), using "leaving-one-out" evaluation, I get an accuracy of 41%

Beginning search

Using feature(s) {1} accuracy is 76%
Using feature(s) {2} accuracy is 79%
Using feature(s) {3} accuracy is 80%
Using feature(s) {4} accuracy is 71%
Using feature(s) {5} accuracy is 68%
Using feature(s) {6} accuracy is 68%
Using feature(s) {7} accuracy is 76%
Using feature(s) {8} accuracy is 83%
Using feature(s) {9} accuracy is 74%
Using feature(s) {10} accuracy is 70%

Feature set {8} was best, accuracy is 83%

Using feature(s) {8,1} accuracy is 78%
Using feature(s) {8,2} accuracy is 93%
Using feature(s) {8,3} accuracy is 82%
Using feature(s) {8,4} accuracy is 81%
Using feature(s) {8,5} accuracy is 83%
Using feature(s) {8,6} accuracy is 83%
Using feature(s) {8,7} accuracy is 83%
Using feature(s) {8,9} accuracy is 73%
Using feature(s) {8,10} accuracy is 84%

Feature set {8,2} was best, accuracy is 93%

Using feature(s) {8,2,1} accuracy is 95%
Using feature(s) {8,2,3} accuracy is 88%
Using feature(s) {8,2,4} accuracy is 87%
Using feature(s) {8,2,5} accuracy is 86%
Using feature(s) {8,2,6} accuracy is 87%
Using feature(s) {8,2,7} accuracy is 86%
Using feature(s) {8,2,9} accuracy is 91%
Using feature(s) {8,2,10} accuracy is 91%

Feature set {8,2,1} was best, accuracy is 95%

Using feature(s) {8,2,1,3} accuracy is 91%
Using feature(s) {8,2,1,4} accuracy is 86%
Using feature(s) {8,2,1,5} accuracy is 83%
Using feature(s) {8,2,1,6} accuracy is 88%
Using feature(s) {8,2,1,7} accuracy is 87%
Using feature(s) {8,2,1,9} accuracy is 87%
Using feature(s) {8,2,1,10} accuracy is 90%

(Warning, Accuracy has decreased!)
Finished Search, best feature subset is {8,2,1}, which has an accuracy of 95%
```