CS170

Project 2:Feature Selection with nearest neighbour

Student Name: Jingfang Guan SID:862218775

Solution:

Dataset	Best Feature Set	Accuracy
Small data #30	Forward selection = {9,3,8}	0.95
Small data #30	Backward elimination = {10, 8 ,6, 5, 3, 2}	0.86
Large data #30	Forward selection = {6,2}	0.965
Large data #30	Backward elimination = {40,39,37,36,35,34,33,32,31,30,29,28,27,26,23,20,19 ,18,17,16,15,14,13,12,11,9,7,6,4,2}	0.784

I. Introduction

The program is designed to look through a given dataset and determine the best features set. It then can be used to estimate the class of future items with the best features. To compute the possibilities to correctly classify an item of each feature, the program needs to find each item's features nearest neighbour and if it is correctly classified. The total number of correctly classified classes divided by the total number of data will be the probability.

II. Challenges

The challenges of the program are mostly the program designs in my opinion. Since C++ classification and libraries are a bit different from Python, the value types have to be accurate while calculating the shortest distance. I have caused a bug in my old code simply due to a valuable type of integer changing my classification correctness. Moreover, C++ doesn't have a numpy library, so I wrote a function to loop and calculate the features distance.

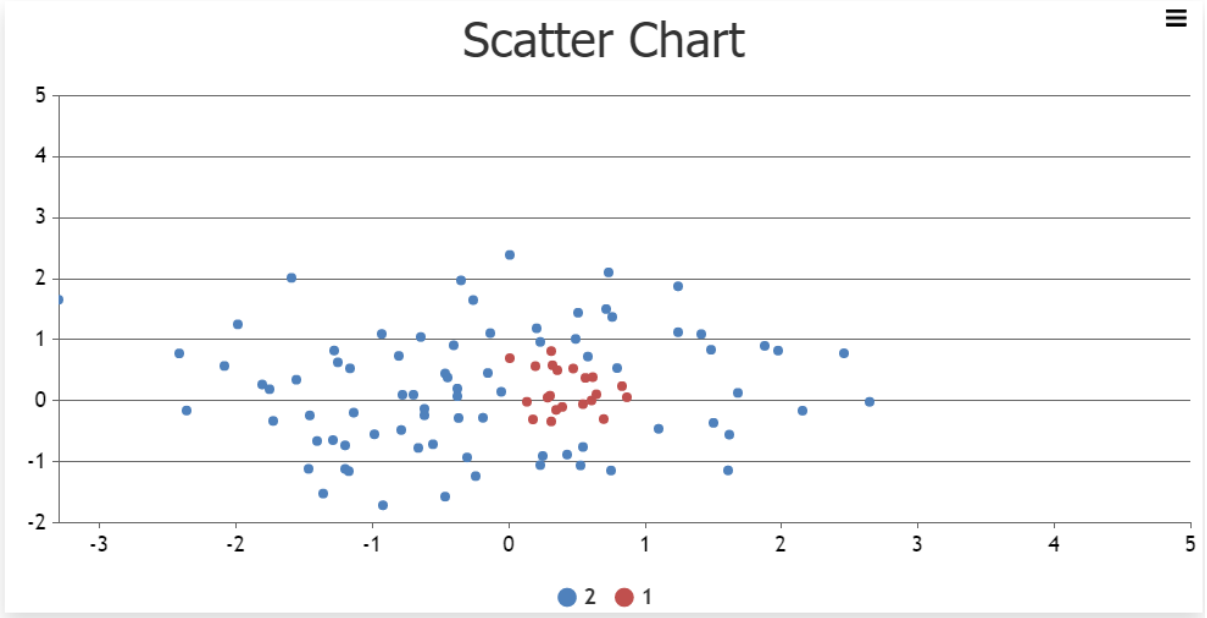
III. Code Design

The code design has been generalised to accept different sizes of dataset, which in a format of first column being the class (in integer) and the columns after it are the features' data. I designed it to store the dataset into a vector of integer vectors. Data[i][0] will be the class label of item i and Data[i][j] will be the value of item i's feature j.

IV. Dataset details

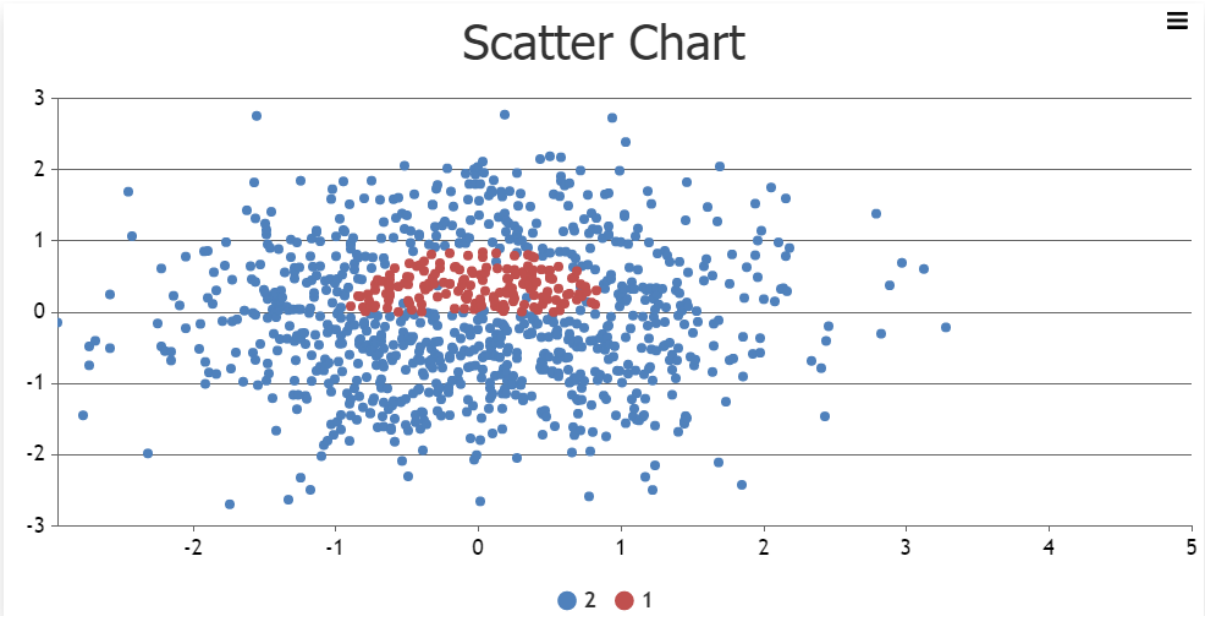
CS170_Spring_2022_Small_data__30.txt: 10 features, 40 instances

X-axis: feature 8, y-axis: feature 3



CS170_Spring_2022_Large_data__30.txt: 40 features. 100 instances

X-axis: feature 6, y-axis: feature 2



V. Algorithm

Forward selection and backward elimination use similar approaches to calculate the possibility of correct classification. Forward checks each new feature from empty initial and adds the best one, while backward checks all features as initial and eliminates one feature to find the best one.

VI. Analysis

Experiment 1: Comparing Forward Selection vs Backward Elimination

Without feature selection, the P(class) will always be the highest number of classes in the dataset. On the other hand, feature selection draws a linear classification and reduces the chance to incorrectly classify an instance based on how identical the feature is.

Forward selection generates the highest accuracy faster than backward elimination, while backward elimination is slower but it has a bigger feature with highest local accuracy than forward.

Backward starts with aff features and eliminates the least relevant feature, but it is generally slower than forward. On the other hand, Forward is faster in computing accuracy but adding new features may make other added features less significant.

Experiment 2: Effect of normalisation

The accuracy might differ between normalisation and without normalisation, since normalisation reduces redundant data.

Experiment 3: Effect of number neighbours (k)

Larger values of k will reduce the influences of noise instances.

VII. Conclusion

After making the program into work, I have realised how much more I will need to practise and learn to make more practical code that can deal with problems with large data. The generalisation part definitely becomes more important. The forward and backward algorithms as well as writing function class to compute matrices are definitely a good practice of my coding structure and refresh my knowledge.

The potential improvements are improving the runtime of matrix distance calculations. Better classification and data organisation can also be done for clearer code and future modification of the code.

VIII. Trace of my dataset

Forward selection with small data set, #30:

```
Using feature(s) {2} accuracy is 67%
Using feature(s) {3} accuracy is 74%
Using feature(s) {4} accuracy is 71%
Using feature(s) {5} accuracy is 73%
Using feature(s) {6} accuracy is 70%
Using feature(s) {7} accuracy is 66%
Using feature(s) {8} accuracy is 85%
Using feature(s) {9} accuracy is 60%
Using feature(s) {10} accuracy is 67%

Feature set {8} was best, accuracy is 85%

Using feature(s) {1} accuracy is 73%
Using feature(s) {2} accuracy is 77%
Using feature(s) {3} accuracy is 94%
Using feature(s) {4} accuracy is 71%
Using feature(s) {5} accuracy is 88%
Using feature(s) {6} accuracy is 80%
Using feature(s) {7} accuracy is 81%
Using feature(s) {9} accuracy is 81%
Using feature(s) {10} accuracy is 71%

Feature set {3,8} was best, accuracy is 94%

Using feature(s) {1} accuracy is 92%
Using feature(s) {2} accuracy is 93%
Using feature(s) {4} accuracy is 85%
Using feature(s) {5} accuracy is 92%
Using feature(s) {6} accuracy is 91%
Using feature(s) {7} accuracy is 94%
Using feature(s) {9} accuracy is 95%
Using feature(s) {10} accuracy is 91%

Feature set {9,3,8} was best, accuracy is 95%

Using feature(s) {1} accuracy is 85%
Using feature(s) {2} accuracy is 93%
Using feature(s) {4} accuracy is 83%
Using feature(s) {5} accuracy is 83%
Using feature(s) {6} accuracy is 84%
Using feature(s) {7} accuracy is 92%
Using feature(s) {10} accuracy is 82%

(Warning, accuracy has decreased!)
Finished search!! The best feature sebset is {9,3,8}, which has an accuracy of 95%
```

Forward selection with large data set, #30:

```
Using feature(s) {14} accuracy is 95.3%
Using feature(s) {15} accuracy is 95.2%
Using feature(s) {16} accuracy is 95.5%
Using feature(s) {17} accuracy is 94.1%
Using feature(s) {18} accuracy is 94.3%
Using feature(s) {19} accuracy is 93.9%
Using feature(s) {20} accuracy is 95.3%
Using feature(s) {21} accuracy is 93.9%
Using feature(s) {22} accuracy is 94%
Using feature(s) {23} accuracy is 94.4%
Using feature(s) {24} accuracy is 94.9%
Using feature(s) {25} accuracy is 92.9%
Using feature(s) {26} accuracy is 94.1%
Using feature(s) {27} accuracy is 93.3%
Using feature(s) {28} accuracy is 94%
Using feature(s) {29} accuracy is 94.8%
Using feature(s) {30} accuracy is 92.6%
Using feature(s) {31} accuracy is 94.4%
Using feature(s) {32} accuracy is 95.1%
Using feature(s) {33} accuracy is 94%
Using feature(s) {34} accuracy is 94.2%
Using feature(s) {35} accuracy is 94.4%
Using feature(s) {36} accuracy is 93.2%
Using feature(s) {37} accuracy is 96%
Using feature(s) {38} accuracy is 93.6%
Using feature(s) {39} accuracy is 94.6%
Using feature(s) {40} accuracy is 94.8%

(Warning, accuracy has decreased!)
Finished search!! The best feature sebset is {6,2}, which has an accuracy of 96.5%
```

Backward Elimination with small data set, #30:

```
Using feature(s) {2,3,4,5,6,7,8,9} accuracy is 72%

Feature set {10,9,8,6,5,4,3,2} was best, accuracy is 81%

Using feature(s) {3,4,5,6,8,9,10} accuracy is 76%
Using feature(s) {2,4,5,6,8,9,10} accuracy is 73%
Using feature(s) {2,3,5,6,8,9,10} accuracy is 86%
Using feature(s) {2,3,4,6,8,9,10} accuracy is 77%
Using feature(s) {2,3,4,5,8,9,10} accuracy is 81%
Using feature(s) {2,3,4,5,6,9,10} accuracy is 67%
Using feature(s) {2,3,4,5,6,8,10} accuracy is 84%
Using feature(s) {2,3,4,5,6,8,9} accuracy is 73%

Feature set {10,9,8,6,5,3,2} was best, accuracy is 86%

Using feature(s) {3,5,6,8,9,10} accuracy is 80%
Using feature(s) {2,5,6,8,9,10} accuracy is 76%
Using feature(s) {2,3,6,8,9,10} accuracy is 79%
Using feature(s) {2,3,5,8,9,10} accuracy is 83%
Using feature(s) {2,3,5,6,9,10} accuracy is 71%
Using feature(s) {2,3,5,6,8,10} accuracy is 84%
Using feature(s) {2,3,5,6,8,9} accuracy is 83%

(Warning, accuracy has decreased!)
Finished search!! The best feature sebset is {10,8,6,5,3,2}, which has an accuracy of 86%
```

Backward Elimination with large data set, #30:

```
Using feature(s) {2,4,6,7,9,11,12,13,14,15,16,17,18,19,20,23,24,26,27,28,29,30,31,32,33,35,36,37,39,40} accuracy is 76.5
%
Using feature(s) {2,4,6,7,9,11,12,13,14,15,16,17,18,19,20,23,24,26,27,28,29,30,31,32,33,34,36,37,39,40} accuracy is 76.4
%
Using feature(s) {2,4,6,7,9,11,12,13,14,15,16,17,18,19,20,23,24,26,27,28,29,30,31,32,33,34,35,37,39,40} accuracy is 76%
Using feature(s) {2,4,6,7,9,11,12,13,14,15,16,17,18,19,20,23,24,26,27,28,29,30,31,32,33,34,35,36,39,40} accuracy is 77.4
%
Using feature(s) {2,4,6,7,9,11,12,13,14,15,16,17,18,19,20,23,24,26,27,28,29,30,31,32,33,34,35,36,37,40} accuracy is 75.1
%
Using feature(s) {2,4,6,7,9,11,12,13,14,15,16,17,18,19,20,23,24,26,27,28,29,30,31,32,33,34,35,36,37,39} accuracy is 75.4
%

here
(Warning, accuracy has decreased!)
Finished search!! The best feature sebset is {40,39,37,36,35,34,33,32,31,30,29,28,27,26,23,20,19,18,17,16,15,14,13,12,11
,9,7,6,4,2}, which has an accuracy of 78.4%
```