Project 2: Feature Selection with Nearest Neighbor
CS 205. Introduction to Artificial Intelligence
Yizhuo Instructor: Dr Eamonn Keogh
14-March-2018

In complete this homework, I consulted…

1. *Bank Marketing Data Set, UCI Machine Learning Repository, https://archive.ics.uci.edu/ml/datasets/Bank+Marketing*
2. Eamonn Keogh (2018), *MachineLearning002* [Power Point presentation], *http://www.cs.ucr.edu/~eamonn/205*
3. Eamonn Keogh (2018), *Project_two_evaluation* [Power Point presentation], *http://www.cs.ucr.edu/~eamonn/205*
4. *OpenRefine (2018), OpenRefine Documentation, http://openrefine.org/documentation.html*
5. Student Chun-Yu Chuang, who is a student taking AI class before. I uses his data set and results to verify my program.
6. The MathWorks (2018), *Documentation, https://www.mathworks.com/help/*

All the code is original. Unimportant subroutines that are not completely original are…
1. The “zscore” function which is available at https://www.mathworks.com/help/stats/zscore.html

CS205 Artificial Intelligence Project 2 : Feature Selection

-Overview

This is the second project of CS205 Artificial Intelligence class of 2018 winter . The project requires us to do feature selection with nearest neighbor. Algorithms contains 1) forward selection, 2) backward selection and another optimized search algorithm. The optimization could speed up the searching or make the search more accurate. This project did a speed up tricky in “leave one out validation” and applied it to both forward and backward search algorithms. So, the program contains 3) Forward Search with Pruning and 4) Backward Search with Pruning. The data set used in this projects are “CS205_SMALLtestdata__5.txt” and “CS205_BIGtestdata__19.txt”.

-Section I: Matlab Code: main.m(page 1)

```
function main()
    disp('Welcome to my Feature Selection Algorithm.');
    fileName = input('Type in the name of the file to test :', 's');
    disp('Type the number of the algorithm you want to run.');
    disp('1) Forward Selection')
    disp('2) Backward Elimination')
    disp('3) Forward Search with Pruning (Make Forward Selection faster)')
    choice = input('4) Backward Search with Pruning (Make Backward Selection
faster)');
    data = load(fileName);
    [row, coloumn] = size(data);
    disp(['This dataset has ', num2str(coloumn-1) ' features (not including the
class attribute), with ', num2str(row), ' instances'])
    disp('Please wait while I normalize the data...')
    normData = dataNormalization(data);
    disp('Done!')
    %run all the features to get an accuracy
    allFeatures(1:coloumn - 1)  = 1:1:(coloumn - 1);
    acc = validation(normData, allFeatures,0,0,0);
    disp(['Running nearest neighbor with all ', num2str(coloumn-1),  ' features,
using leaving-one-out evaluation,'])
    disp(['I get an accuracy of ', num2str(acc*100), '%']);
    disp('Beginning search.');
    %0 means with out speed up, 1 means speed up algorithm
    if choice == 1
        forward(normData, 0)
    elseif choice == 2
        backward(normData, 0)
    elseif choice == 3
        forward(normData, 1)
    elseif choice == 4
        backward(normData, 1)
    else
        disp('invalid input, bye~!');
    end
end

function normData = dataNormalization(data)
    [row, coloumn] = size(data);
    featureData = data(:, 2:coloumn);
    [resultData, dataMean, dataStd] = zscore(featureData);
    label = data(:, 1);
    normData = [label, resultData];
end
```

Matlab Code: main.m (page 2)

```
%function @validation: leave_one_out_cross_validation
%arg1@data: the whole normalized traning data with lable.
%arg2@current_set_of_features: the feature No
%arg3@k: feature to add or exclude
%arg4@flag: if flag == 1, we add kth feature, flag == 2, we exclude it
%arg5@acc_so_far: To help speed up the algorithm
function acc = validation(data,current_set_of_features,k, flag, acc_so_far)
    right = 0;
    wrong = 0;
    [row, coloumn] = size(data);
    if k~=0
        if flag == 1
            current_set_of_features = union(current_set_of_features, k);
        elseif flag == 2
            current_set_of_features = setdiff(current_set_of_features, k);
        end
    end
    [frow, fcol] = size(current_set_of_features);
    label = data(:, 1);
    featureData = data(:, 2:coloumn);
    %initialize training data
    tData = [];
    for i = 1 : fcol
        tData = [tData, featureData(:, current_set_of_features(1, i))];
    end
    tData = [label, tData];
    [trow, tcol] = size(tData);
    %leave one out
    for i = 1 : row
        %exclude the row
        testData = tData;
        testData(i, :) = [];
        %the separate the labels and features of the rest training data
        trainLable = testData(:,1);
        trainSet = testData(:, 2:tcol);
        %get features and label of the test data
        test = tData(i, 2:tcol);
        realLabel = tData(i, 1);
       %knn algorithm, k = 1 since it's nearest neighbor
        classLable = knn(test, trainSet, trainLable, 1);
        if classLable == realLabel
            right = right + 1;
        else
            wrong = wrong + 1;
            if wrong > (row * (1 - acc_so_far))
               acc = 0;
               return;
            end
        end
    end
    acc = right/(right + wrong);
end

function classLabel = knn(test, data, labels, k)
    [row , col] = size(data);
    diffMat = repmat(test,[row,1]) - data ;
    distanceMat = sqrt(sum(diffMat.^2,2));
    [B , IX] = sort(distanceMat,'ascend');
    len = min(k,length(B));
    classLabel = mode(labels(IX(1:len)));
end
```

Matlab Code: main.m (page 3)

```
%arg1@data: the whole normalized traning data with lable.
%arg2@opt: opt = 0 means no optimization, opt = 1 means speed up algorithm.
function forward(data,opt)
    current_set_of_features = []; % Initialize an empty set
    optimal_set_of_features = []; % Store the optimal result
    optimal_accuracy = 0;

    %Comparing the time, we use tic and toc to record the searching time.
    tic;
    t1 = clock;
    for i = 1 : size(data,2)-1
        %%disp(['On the ',num2str(i),'th level of the search tree'])
        feature_to_add_at_this_level = [];
        best_so_far_accuracy = 0;

        for k = 1 : size(data,2)-1
            if isempty(intersect(current_set_of_features,k))
                if opt == 0
                    %leave_one_out_cross_validation
                    accuracy = validation(data,current_set_of_features,k, 1, 0);
                elseif opt == 1
                    accuracy = validation(data,current_set_of_features,k, 1,
                                  best_so_far_accuracy);
                end
                %disp(['        Using feature(s){ ',num2str(current_set_of_features),
                 %'  ', num2str(k),' }', 'accuracy is ', num2str(100*accuracy), '%']);

                if accuracy > best_so_far_accuracy
                    best_so_far_accuracy = accuracy;
                    feature_to_add_at_this_level = k;
                end
            end
        end
        %disp(['On level ', num2str(i),' I added feature ',
             %num2str(feature_to_add_at_this_level), ' to current set']);
        current_set_of_features = [current_set_of_features,
                                   feature_to_add_at_this_level];
        disp(['Feature set {', num2str(current_set_of_features), ' } was best,
             accuracy is ', num2str(100*best_so_far_accuracy), '%']);

        if best_so_far_accuracy > optimal_accuracy
            optimal_accuracy = best_so_far_accuracy;
            optimal_set_of_features = current_set_of_features;
        else
            disp('(Warning, Accuracy has decreased! Continuing search
                    in case of local maxima)');
        end
    end
    toc;
    disp(['Finished search!! The best feature subset is {',
              num2str(optimal_set_of_features), '}, which has an accuracy of
              ',num2str(100*optimal_accuracy), '%']);

end
```

Matlab Code: main.m (page 4)

```
%arg1@data: the whole normalized traning data with lable.
%arg2@opt: opt = 0 means no optimization, opt = 1 means speed up algorithm.
function backward(data, opt)
    [row, coloumn] = size(data);
    current_set_of_features(1:coloumn - 1)  = 1:1:(coloumn - 1); % Initialize a full
feature set
    optimal_set_of_features = []; % Store the optimal result
    optimal_accuracy = 0;
    tic;
    t1 = clock;
    for i = 1 : size(data,2)-1
        disp(['On the ',num2str(i),'th level of the search tree'])
        feature_to_exclude_at_this_level = [];
        best_so_far_accuracy = 0;
        best_so_far_features = [];

        for k = 1 : size(data,2)-1
            if ~isempty(intersect(current_set_of_features,k))
                if opt == 0
                    accuracy = validation(data,current_set_of_features,k,2, 0);
                elseif opt == 1
                    accuracy = validation(data,current_set_of_features,k,2,
best_so_far_accuracy);
                end
                disp(['Using feature(s){ ',num2str(setdiff(current_set_of_features,
k)),' }', 'accuracy is ', num2str(100*accuracy), '%']);

                if accuracy > best_so_far_accuracy
                    best_so_far_accuracy = accuracy;
                    feature_to_exclude_at_this_level = k;
                    best_so_far_features = setdiff(current_set_of_features, k);
                end
            end
        end
        disp(['best_so_far_features: ', num2str(best_so_far_features),
'best_so_far_accuracy = ', num2str(best_so_far_accuracy)]);
        disp(['On level ', num2str(i),' I excluded feature ',
num2str(feature_to_exclude_at_this_level), ' to current set']);

        current_set_of_features = setdiff(current_set_of_features,
feature_to_exclude_at_this_level);
        if best_so_far_accuracy > optimal_accuracy
            optimal_accuracy = best_so_far_accuracy;
            optimal_set_of_features = best_so_far_features;
        else
            disp('(Warning, Accuracy has decreased! Continuing search in case of local
maxima)');
        end
    end
    toc;
    disp(['Finished search!! The best feature subset is {',
num2str(optimal_set_of_features), '}, which has an accuracy of
',num2str(100*optimal_accuracy), '%']);
end
```

Section II: Test Results and Explanations

1-1. Forward Selection on “CS205_SMALLtestdata__5.txt”, the best feature subset is {6 8}, which has an accuracy of 93%.

```
Welcome to my Feature Selection Algorithm.
Type in the name of the file to test :CS205_SMALLtestdata__5.txt
Type the number of the algorithm you want to run.
1) Forward Selection
2) Backward Elimination
3) Forward Pruning (Make Forward Selection faster)
4) Backward Pruning (Make Backward Selection faster)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 all 10 features, using leaving-one-out evaluation,
I get an accuracy of 69%
Beginning search.
Feature set {6 } was best, accuracy is 83%
Feature set {6  8 } was best, accuracy is 93%
Feature set {6  8  3 } was best, accuracy is 85%
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Feature set {6  8  3  2 } was best, accuracy is 83%
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Feature set {6  8  3  2  1 } was best, accuracy is 82%
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Feature set {6  8  3  2  1  7 } was best, accuracy is 82%
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Feature set {6  8  3  2  1  7  4 } was best, accuracy is 77%
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Feature set {6  8  3  2  1  7  4  9 } was best, accuracy is 74%
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Feature set {6  8  3  2  1  7  4  9  10 } was best, accuracy is 73%
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Feature set {6  8  3  2  1  7  4  9  10  5 } was best, accuracy is 69%
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Elapsed time is 0.420 seconds.
Finished search!! The best feature subset is {6  8}, which has an accuracy of 93%
```

1-2. Forward Selection of "CS205_BIGtestdata__19.txt", the best feature subset is {19 48 42}, which has an accuracy of 92%

```
**********Introduction Omitted **********
This dataset has 50 features (not including the class attribute), with 100 instances
Please wait while I normalize the data...
Done!
Running nearest neighbor with all 50 features, using leaving-one-out evaluation,
I get an accuracy of 73%
Beginning search.
Feature set {19 } was best, accuracy is 84%
Feature set {19 48 } was best, accuracy is 85%
Feature set {19 48 42 } was best, accuracy is 92%
Feature set {19 48 42 36 } was best, accuracy is 91%
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Feature set {19 48 42 36 40 } was best, accuracy is 90%
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Feature set {19 48 42 36 40 39 } was best, accuracy is 89%
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Feature set {19 48 42 36 40 39 20 } was best, accuracy is 89%
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Feature set {19 48 42 36 40 39 20 6 } was best, accuracy is 88%
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Feature set {19 48 42 36 40 39 20 6 7 } was best, accuracy is 89%
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Feature set {19 48 42 36 40 39 20 6 7 14 } was best, accuracy is 89%
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Feature set {19 48 42 36 40 39 20 6 7 14 9 } was best, accuracy is 89%
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Feature set {19 48 42 36 40 39 20 6 7 14 9 10 } was best, accuracy is 89%
**********Some Output Omitted**********
Feature set {19 48 42 36 40 39 20 6 7 14 9 10 11 30 33 22 5 13 15 8 43 25 35 34
18 37 27 1 38 3 28 23 24 41 44 17 31 49 4 2 47 16 50 32 45 12 21 } was best,
accuracy is 83%
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Feature set {19 48 42 36 40 39 20 6 7 14 9 10 11 30 33 22 5 13 15 8 43 25 35 34
18 37 27 1 38 3 28 23 24 41 44 17 31 49 4 2 47 16 50 32 45 12 21 29 } was best,
accuracy is 82%
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Feature set {19 48 42 36 40 39 20 6 7 14 9 10 11 30 33 22 5 13 15 8 43 25 35 34
18 37 27 1 38 3 28 23 24 41 44 17 31 49 4 2 47 16 50 32 45 12 21 29 26 } was
best, accuracy is 78%
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Feature set {19 48 42 36 40 39 20 6 7 14 9 10 11 30 33 22 5 13 15 8 43 25 35 34
18 37 27 1 38 3 28 23 24 41 44 17 31 49 4 2 47 16 50 32 45 12 21 29 26 46 }
was best, accuracy is 73%
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Elapsed time is 5.916 seconds.
Finished search!! The best feature subset is {19 48 42}, which has an accuracy of 92%
```

2-1. Backward Elimination of "CS205_SMALLtestdata__5.txt", the best feature subset is {6 8}, which has an accuracy of 93%

```
**********Introduction Omitted **********
Running nearest neighbor with all 10 features, using leaving-one-out evaluation,
I get an accuracy of 69%
Beginning search.
Feature set {1 2 3 4 5 6 8 9 10} was best, accuracy is 0.75
Feature set {1 2 3 5 6 8 9 10} was best, accuracy is 0.76
Feature set {1 2 5 6 8 9 10} was best, accuracy is 0.78
Feature set {1 2 5 6 8 9} was best, accuracy is 0.82
Feature set {2 5 6 8 9} was best, accuracy is 0.83
Feature set {5 6 8 9} was best, accuracy is 0.83
Feature set {5 6 8} was best, accuracy is 0.85
Feature set {6 8} was best, accuracy is 0.93
Feature set {6} was best, accuracy is 0.83
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Feature set {} was best, accuracy is 0.79
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Elapsed time is 0.431 seconds.
Finished search!! The best feature subset is {6 8}, which has an accuracy of 93%
```

2-2. Backward Elimination of "CS205_BIGtestdata__19.txt", the best feature subset is {2 6 11 14 15 21 22 23 24 25 28 30 34 41 42 43 44 45 47 49 50}, which has an accuracy of 91%.

```
Running nearest neighbor with all 50 features, using leaving-one-out evaluation,
I get an accuracy of 73%
Beginning search.
Feature set {1 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 31 32 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50}
was best, accuracy is 0.79
Feature set {1 2 3 4 5 6 7 8 9 10 11 13 14 15 16 17 18 19 20 21 22 23 24 25
26 27 28 29 30 31 32 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50} was
best, accuracy is 0.84
**********Some Output Omitted**********
Feature set {2 6 11 14 15 21 22 23 24 25 28 30 34 41 42 43 44 45 47 49 50} was
best, accuracy is 0.91
Feature set {2 6 11 14 15 21 22 24 25 28 30 34 41 42 43 44 45 47 49 50} was best,
accuracy is 0.9
**********Some Output Omitted**********
Feature set {1 2 3 4 5 6 7 8 9 10 11 14 15 16 17 18 19 21 22 23 24 25 26 27
28 30 31 32 34 35 36 37 38 39 40 41 42 43 44 45 47 48 49 50} was best, accuracy is
0.86
Feature set {1 2 3 4 5 6 8 9 10 11 14 15 16 17 18 19 21 22 23 24 25 26 27 28
30 31 32 34 35 36 37 38 39 40 41 42 43 44 45 47 48 49 50} was best, accuracy is 0.86
Feature set {1 2 3 4 6 8 9 10 11 14 15 16 17 18 19 21 22 23 24 25 26 27 28 30
31 32 34 35 36 37 38 39 40 41 42 43 44 45 47 48 49 50} was best, accuracy is 0.87
**********Some Output Omitted**********
Elapsed time is 7.148 seconds.
Finished search!! The best feature subset is {2 6 11 14 15 21 22 23 24 25 28 30 34 41 42
43 44 45 47 49 50}, which has an accuracy of 91%
```

3-1. Forward Selection with Pruning on “CS205_SMALLtestdata__5.txt”, the best feature subset is {6 8}, which has an accuracy of 93%. The result is the same as unoptimized algorithm, but with the same line of output, the time use drop from 0.420 seconds to 0.385 seconds (8.33%). The timer function is “tic” and “toc” from matlab.

```
Welcome to my Feature Selection Algorithm.
Type in the name of the file to test :CS205_SMALLtestdata__5.txt
Type the number of the algorithm you want to run.
1) Forward Selection
2) Backward Elimination
3) Forward Pruning (Make Forward Selection faster)
4) Backward Pruning (Make Backward Selection faster)3
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 all 10 features, using leaving-one-out evaluation,
I get an accuracy of 69%
Beginning search.
Feature set {6 } was best, accuracy is 83%
Feature set {6  8 } was best, accuracy is 93%
Feature set {6  8  3 } was best, accuracy is 85%
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Feature set {6  8  3  2 } was best, accuracy is 83%
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Feature set {6  8  3  2  1 } was best, accuracy is 82%
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Feature set {6  8  3  2  1  7 } was best, accuracy is 82%
Feature set {6  8  3  2  1  7  4 } was best, accuracy is 77%
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Feature set {6  8  3  2  1  7  4  9 } was best, accuracy is 74%
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Feature set {6  8  3  2  1  7  4  9  10 } was best, accuracy is 73%
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Feature set {6  8  3  2  1  7  4  9  10  5 } was best, accuracy is 69%
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Elapsed time is 0.385 seconds.
Finished search!! The best feature subset is {6  8}, which has an accuracy of 93%
```

Function “leave_one_out_cross_validation” calculates the accuracy of nearest neighbor classifier with given features. In each level of the search tree, if the feature being evaluated now owns lower accuracy than adding the features who has been evaluated, then the calculation can stop here.

For example, “CS205_SMALLtestdata__5.txt” offers 10 features, and the following shows the details of unoptimized search.

```
On the 1th level of the search tree
      Using feature(s){   1 }accuracy is 66%
      Using feature(s){   2 }accuracy is 63%
      Using feature(s){   3 }accuracy is 63%
      Using feature(s){   4 }accuracy is 66%
      Using feature(s){   5 }accuracy is 63%
      Using feature(s){   6 }accuracy is 83%
      Using feature(s){   7 }accuracy is 71%
      Using feature(s){   8 }accuracy is 76%
      Using feature(s){   9 }accuracy is 67%
      Using feature(s){   10 }accuracy is 59%
On level 1 I added feature 6 to current set
Feature set {6 } was best, accuracy is 83%
```

The accuracy of feature 1 is calculated and recorded as the best accuracy so far. When feature 2 is evaluated, there will be point that over (1 - 66%) = 34% labels are predicted incorrectly. At this time, the validation function can stop and return 0 since feature 2 will not perform better than feature 1. It's a waste of time to calculate how bad it is. This process sounds like alpha-beta pruning in minimax algorithm, therefore, I put words "pruning" in its name.
After the optimization, the detail of the optimized forward selection is:

```
**********Introduction Omitted **********
Running nearest neighbor with all 10 features, using leaving-one-out evaluation,
I get an accuracy of 69%
Beginning search.
On the 1th level of the search tree
      Using feature(s){   1 }accuracy is 66%
      Using feature(s){   2 }accuracy is 0%
      Using feature(s){   3 }accuracy is 0%
      Using feature(s){   4 }accuracy is 66%
      Using feature(s){   5 }accuracy is 0%
      Using feature(s){   6 }accuracy is 83%
      Using feature(s){   7 }accuracy is 0%
      Using feature(s){   8 }accuracy is 0%
      Using feature(s){   9 }accuracy is 0%
      Using feature(s){   10 }accuracy is 0%
On level 1 I added feature 6 to current set
Feature set {6 } was best, accuracy is 83%
On the 2th level of the search tree
      Using feature(s){ 6  1 }accuracy is 72%
      Using feature(s){ 6  2 }accuracy is 74%
**********Some Output Omitted**********
On the 9th level of the search tree
      Using feature(s){ 6  8  3  2  1  7  4  9  5 }accuracy is 72%
      Using feature(s){ 6  8  3  2  1  7  4  9  10 }accuracy is 73%
On level 9 I added feature 10 to current set
Feature set {6   8   3   2   1   7   4   9  10 } was best, accuracy is 73%
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
On the 10th level of the search tree
      Using feature(s){ 6   8   3   2   1   7   4   9  10  5 }accuracy is 69%
On level 10 I added feature 5 to current set
Feature set {6   8   3   2   1   7   4   9  10   5 } was best, accuracy is 69%
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Finished search!! The best feature subset is {6  8}, which has an accuracy of 93%
```

3-2. Forward Selection with Pruning on “CS205_BIGtestdata__19.txt”, the best feature subset is {19 48 42}, which has an accuracy of 92%. The result is the same as unoptimized algorithm, but with the same line of output, the time use drop from 5.915 seconds to 4.876 seconds (17.56%).

```
**********Introduction Omitted **********
Beginning search.
Feature set {19 } was best, accuracy is 84%
Feature set {19  48 } was best, accuracy is 85%
Feature set {19  48  42 } was best, accuracy is 92%
Feature set {19  48  42  36 } was best, accuracy is 91%
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Feature set {19  48  42  36  40 } was best, accuracy is 90%
(Warning, Accuracy has decreased! Continuing search in case of local maxima)

**********Some Output Omitted**********

Feature set {19  48  42  36  40  39  20   6   7  14   9  10  11  30  33  22   5  13  15   8  43  25
35  34  18  37  27   1  38   3  28  23  24  41  44  17  31  49   4   2  47  16  50  32  45  12  21
29  26  46 } was best, accuracy is 73%
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Elapsed time is 4.877 seconds.
Finished search!! The best feature subset is {19  48  42}, which has an accuracy of 92%
```

4-1. Backward Elimination with Pruning on “CS205_SMALLtestdata__5.txt”, the best feature subset is {6 8}, which has an accuracy of 93%. The result is the same as unoptimized algorithm, but with the same line of output, the time use drop from 0.431 seconds to 0.355 seconds (17.63%).

```
**********Introduction Omitted **********
Running nearest neighbor with all 10 features, using leaving-one-out evaluation,
I get an accuracy of 69%
Beginning search.
Feature set {1   2   3   4   5   6   8   9  10} was best, accuracy is  0.75
Feature set {1   2   3   5   6   8   9  10} was best, accuracy is  0.76
Feature set {1   2   5   6   8   9  10} was best, accuracy is  0.78
Feature set {1  2  5  6  8  9} was best, accuracy is  0.82
Feature set {2  5  6  8  9} was best, accuracy is  0.83
Feature set {5  6  8  9} was best, accuracy is  0.83
Feature set {5  6  8} was best, accuracy is  0.85
Feature set {6  8} was best, accuracy is  0.93
Feature set {6} was best, accuracy is  0.83
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Feature set {} was best, accuracy is  0.79
(Warning, Accuracy has decreased! Continuing search in case of local maxima)
Elapsed time is 0.355 seconds.
Finished search!! The best feature subset is {6  8}, which has an accuracy of 93%
```

4-2. Backward Elimination with Pruning on “CS205_BIGtestdata__19.txt”, the best feature subset is {2 6 11 14 15 21 22 23 24 25 28 30 34 41 42 43 44 45 47 49 50}, which has an accuracy of 91%. The result is the same as unoptimized algorithm, but with the same line of output, the time use drop from 7.148 seconds to 6.276 seconds (12.20%).

```
**********Introduction Omitted **********
Running nearest neighbor with all 50 features, using leaving-one-out evaluation,
I get an accuracy of 73%
Beginning search.
Feature set {1 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 31 32 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50} was
best, accuracy is 0.79
Feature set {1 2 3 4 5 6 7 8 9 10 11 13 14 15 16 17 18 19 20 21 22 23 24 25
26 27 28 29 30 31 32 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50} was
best, accuracy is 0.84
Feature set {1 2 3 4 5 6 7 8 9 10 11 14 15 16 17 18 19 20 21 22 23 24 25 26
27 28 29 30 31 32 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50} was best,
accuracy is 0.84
Feature set {1 2 3 4 5 6 7 8 9 10 11 14 15 16 17 18 19 20 21 22 23 24 25 26
27 28 30 31 32 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50} was best,
accuracy is 0.84

**********Some Output Omitted**********
Elapsed time is 6.276 seconds.
Finished search!! The best feature subset is {2 6 11 14 15 21 22 23 24 25 28 30 34 41 42
43 44 45 47 49 50}, which has an accuracy of 91%
```

The following table shows the time improvement if all the intermediate results are not printed.

	Forward, Small	Forward, Big	Backward, Small	Backward, Big
optimized	0.225	3.996	0.214	5.185
unoptimized	0.263	4.388	0.264	6.252
improvements	14.44%	8.93%	18.94%	17.07%

Section III: Research on UCI Machine Learning Repository

Overview:

This research applied the forward and backward search algorithm to “Bank Marketing Data Set” from UCI machine learning repository, available: https://archive.ics.uci.edu/ml/datasets/Bank+Marketing. It owns 16 features and a label. Features are used to predict the success of telemarketing calls fro selling bank long-term deposits. There are two datasets involved, a full version with 45221 instances and a short version of 4521 instances, which randomly selected from full version.

Features used are listed below, the categorical attributes such as “job” can be easily transformed to the binary features “isUnemployed”, “isStudent”, etc.

Original features in the data set:
1 - age (numeric)
2 - job : type of job (categorical:
"admin.","unknown","unemployed","management","housemaid","entrepreneur","student",
"blue-collar","self-employed","retired","technician","services")
3 - marital : marital status (categorical: "married","divorced","single"; note: "divorced" means divorced or widowed)
4 - education (categorical: "unknown","secondary","primary","tertiary")
5 - default: has credit in default? (binary: "yes","no")
6 - balance: average yearly balance, in euros (numeric)
7 - housing: has housing loan? (binary: "yes","no")
8 - loan: has personal loan? (binary: “yes","no")
related with the last contact of the current campaign:
9 - contact: contact communication type (categorical: "unknown","telephone","cellular")
10 - day: last contact day of the month (numeric)
11 - month: last contact month of year (categorical: "jan", "feb", "mar", ..., "nov", "dec")
12 - duration: last contact duration, in seconds (numeric)
other attributes:
13 - campaign: number of contacts performed during this campaign and for this client (numeric, includes last contact)
14 - pdays: number of days that passed by after the client was last contacted from a previous campaign (numeric, -1 means client was not previously contacted)
15 - previous: number of contacts performed before this campaign and for this client (numeric)
16 - poutcome: outcome of the previous marketing campaign (categorical:
"unknown","other","failure","success")

The true features used in tests:

feature 1	feature 2	feature 3	feature 4	feature 5	feature 6	feature 7	feature 8
age >=60	age[50, 60)	age[40,50)	age[30,40)	age <30	unemployed	isStudent	isBlueCollar
feature 9	feature 10	feature 11	feature 12	feature 13	feature 14	feature 15	feature 16
isHouseMaid	isServices	isSelfEmployed	isRetired	isTechnician	isAdmin	isManagement	isEntrepreneur
feature 17	feature 18	feature 19	feature 20	feature 21	feature 22	feature 23	feature 24
single	divorced	married	secondaryEdu	primaryEdu	TertiaryEdu	default	balance < 0
feature 25	feature 26	feature 27	feature 28	feature 29	feature 30	feature 31	feature 32
balance[0, 3000)	balance[3000, 6000)	balance[6000, 9000]	balance >9000	housing	loan	contact	day
feature 33	feature 34	feature 35	feature 36	feature 37	feature 38		
month	duration	campaign	pdays	previous	poutcome		

Running the **forward search algorithm with pruning**, the best feature subset is {38 28   8}, which has an accuracy of 89.41%.

Result of forward search algorithm:
This dataset has 38 features (not including the class attribute), with 4521 instances
Please wait while I normalize the data...
Done!
Running nearest neighbor with all 38 features, using leaving-one-out evaluation,
I get an accuracy of 85.58%
Beginning search.
……
Finished search!! The best feature subset is {38 28   8}, which has an accuracy of 89.41%

Running the **backward search algorithm with pruning**, the best feature subset is {38}, which has an accuracy of 89.29%. No new good feature is found in the backward searching.

Running nearest neighbor with all 38 features, using leaving-one-out evaluation,
I get an accuracy of 85.58%
Beginning search.
……
Finished search!! The best feature subset is {38}, which has an accuracy of 89.29%

Findings:
Outcome of the previous marketing campaign is a strongly indicator of the success of telemarketing calls for selling bank long-term deposit. This means the people who accept the last selling is more likely to accept the further selling from telephone call. That makes sense, the people who dislike the selling from telephone is more probable to hang up the phone as soon as they realize who is the caller. This interrupts the further communication, so, the bank staff cannot introduce the product.

isBlueCollar was selected as a good feature in forward search. BlueCollars tends not to subscribe long-term deposit.This may have an explanation but not strong. Two assumptions are considered. First, blue collars may make less money than others, but this may not true in reality. Second, blue collars need money any time, therefore, they cannot leave some money untouched for a long time.

Balance > 9000 was selected as a good feature in forward search. This makes sense, because people who owns high yearly balance is more likely to have long-term deposit.