Project II briefing

Feature Selection with Nearest Neighbor

I am going to give you lots of help, including writing some “guide code”

My “guide code” is MATLAB, which is very close to pseudocode.

Project II briefing

- Revisit slides on Nearest Neighbor Classification

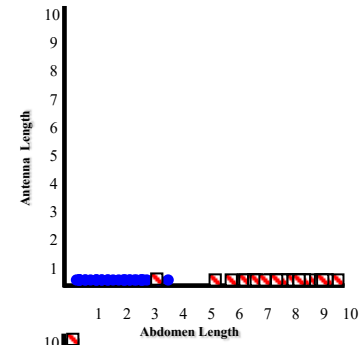
Recall our insect example…

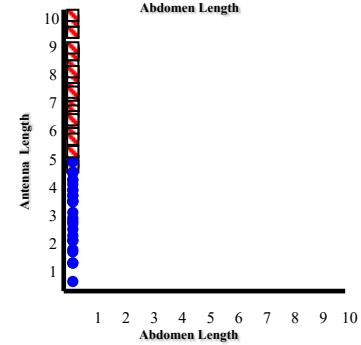
Suppose that we want to build a classifier for it.
We were given two features Abdomen length and Antenna length, but we don't have to use both.
We could use…

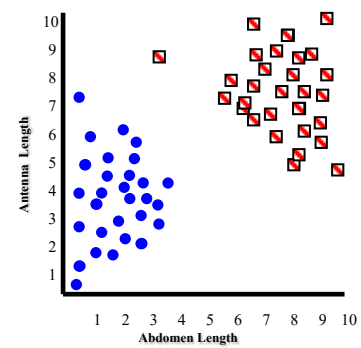
- Abdomen length only
- Antenna length only
- Both Abdomen length and Antenna length

In this case, we can just try all three possible and pick the best.
However, suppose we had Abdomen length and Antenna length, Thorax length, total length, pit diameter , humeral ridge diameter, trochanter length, head height…

How can we pick the best features, when there are say 100s of them?

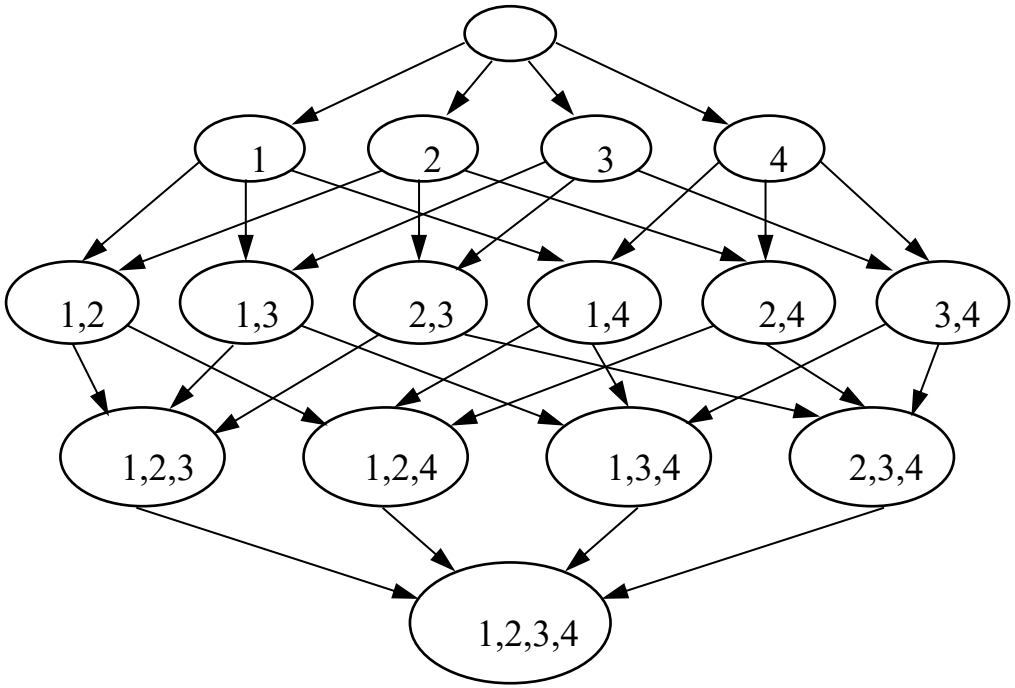






More generally the problem is: Given N features, how do you select the best subset?

For concreteness, in my examples, I will assume N = 4.



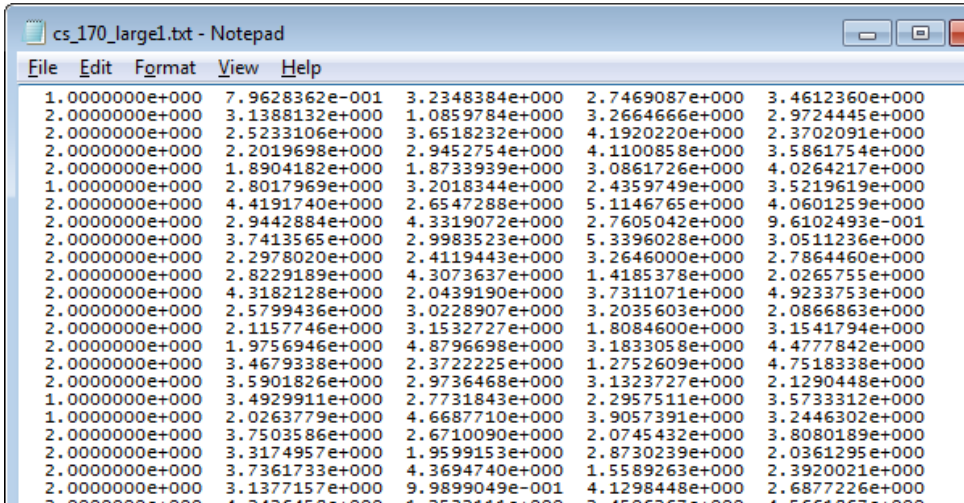
It might be that the problem is:

```
Lung_Cancer |
Not_Lung_Cancer
```

And the features are

1. Height
2. Years working in coal mine
3. Weight
4. Cigarettes per day

To be clear, you will just see files like the one on the right..

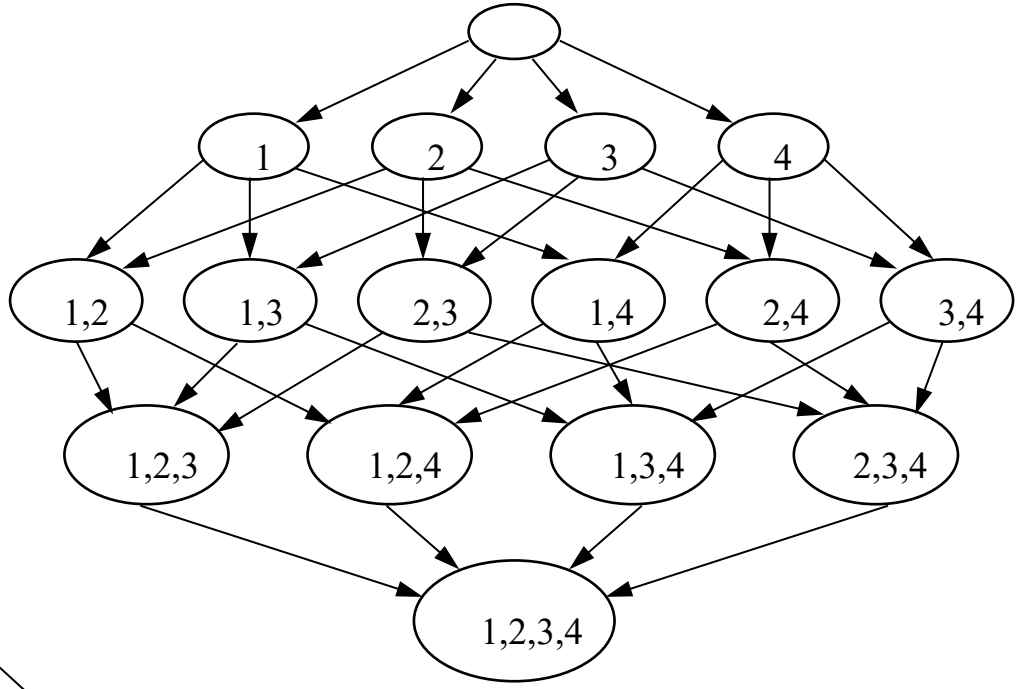


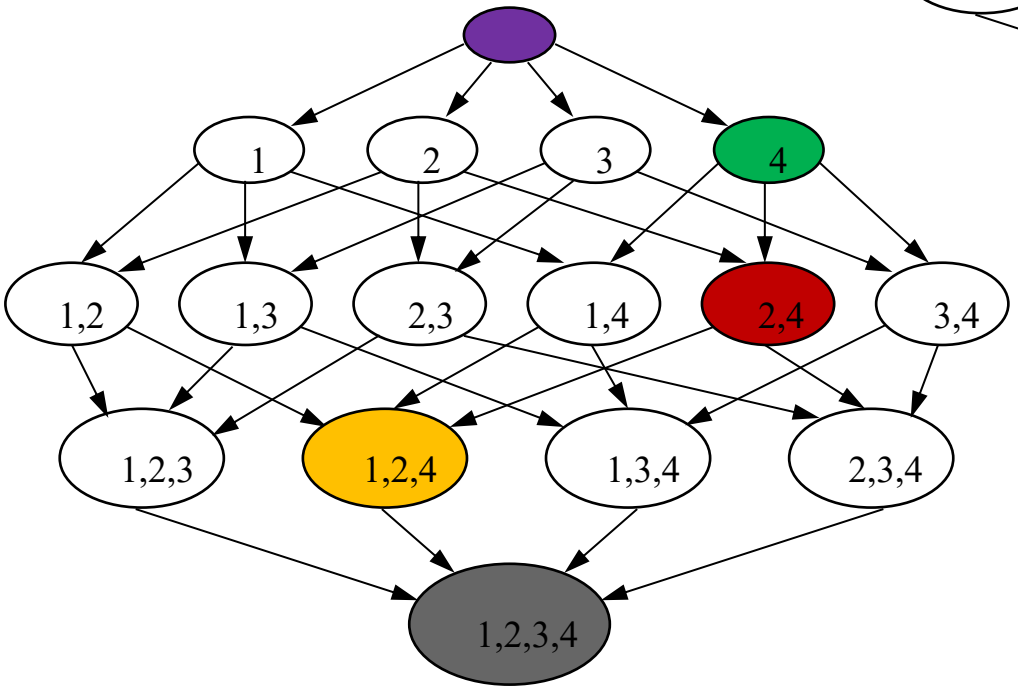
cs_170_large1.txt - Notepad

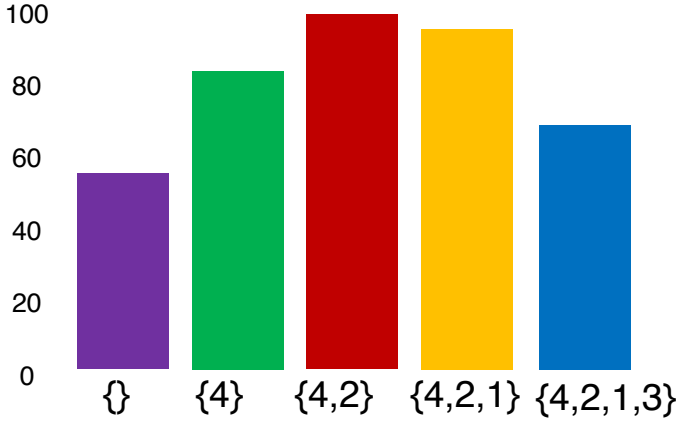
File Edit Format View Help

1.0000000e+000	7.9628362e-001	3.2348384e+000	2.7469087e+000	3.4612360e+000
2.0000000e+000	3.1388132e+000	1.0859784e+000	3.2664666e+000	2.9724445e+000
2.0000000e+000	2.5233106e+000	3.6518232e+000	4.1920220e+000	2.3702091e+000
2.0000000e+000	2.2019698e+000	2.9452754e+000	4.1100858e+000	3.5861754e+000
2.0000000e+000	1.8904182e+000	1.8733939e+000	3.0861726e+000	4.0264217e+000
1.0000000e+000	2.8017969e+000	3.2018344e+000	2.4359749e+000	3.5219619e+000
2.0000000e+000	4.4191740e+000	2.6547288e+000	5.1146765e+000	4.0601259e+000
2.0000000e+000	2.9442884e+000	4.3319072e+000	2.7605042e+000	9.6102493e-001
2.0000000e+000	3.7413565e+000	2.9983523e+000	5.3396028e+000	3.0511236e+000
2.0000000e+000	2.2978020e+000	2.4119443e+000	3.2646000e+000	2.7864460e+000
2.0000000e+000	2.8229189e+000	4.3073637e+000	1.4185378e+000	2.0265755e+000
2.0000000e+000	4.3182128e+000	2.0439190e+000	3.7311071e+000	4.9233753e+000
2.0000000e+000	2.5799436e+000	3.0228907e+000	3.2035603e+000	2.0866863e+000
2.0000000e+000	2.1157746e+000	3.1532727e+000	1.8084600e+000	3.1541794e+000
2.0000000e+000	1.9756946e+000	4.8796698e+000	3.1833058e+000	4.4777842e+000
2.0000000e+000	3.4679338e+000	2.3722225e+000	1.2752609e+000	4.7518338e+000
2.0000000e+000	3.5901826e+000	2.9736468e+000	3.1323727e+000	2.1290448e+000
1.0000000e+000	3.4929911e+000	2.7731843e+000	2.2957511e+000	3.5733312e+000
1.0000000e+000	2.0263779e+000	4.6687710e+000	3.9057391e+000	3.2446302e+000
2.0000000e+000	3.7503586e+000	2.6710090e+000	2.0745432e+000	3.8080189e+000
2.0000000e+000	3.3174957e+000	1.9599153e+000	2.8730239e+000	2.0361295e+000
2.0000000e+000	3.7361733e+000	4.3694740e+000	1.5589263e+000	2.3920021e+000
2.0000000e+000	3.1377157e+000	9.9899049e-001	4.1298448e+000	2.6877226e+000

More generally the problem is: Given N features, how do you select the best subset? For concreteness, in my examples, I will assume N = 4.



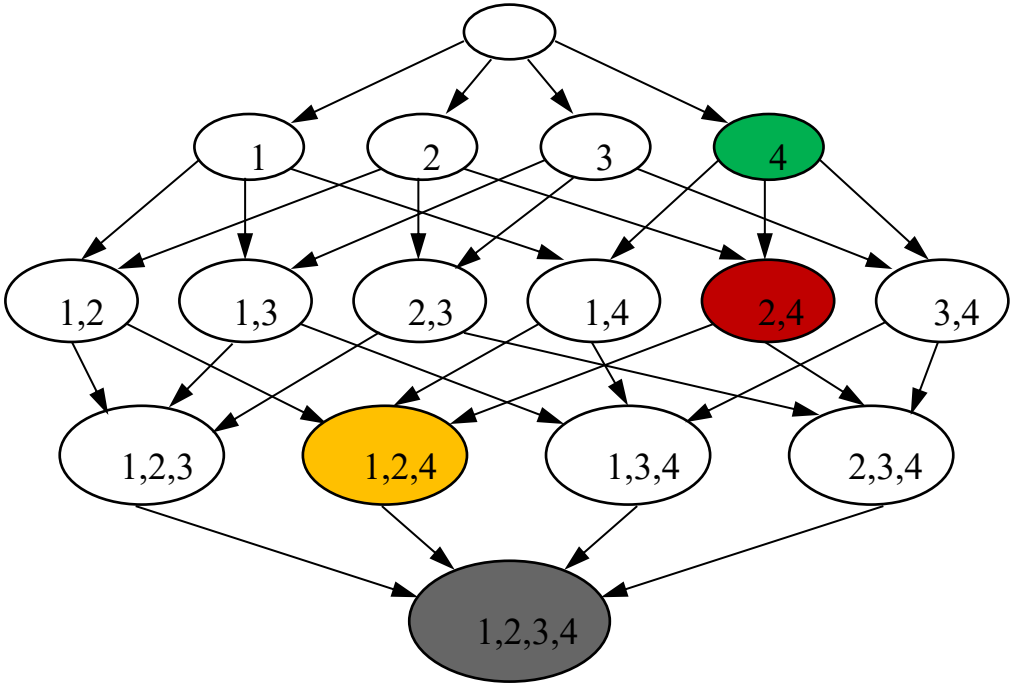




This is what Project 2 Part 1 search “looks” like.
I just want the printout, the figure is for your ref only.

(I also should have printed out the accuracy at each step)

```
EDU>> feature_search_demo(mydata)
On the 1th level of the search tree
--Considering adding the 1 feature
--Considering adding the 2 feature
--Considering adding the 3 feature
--Considering adding the 4 feature
On level 1 i added feature 4 to current set
On the 2th level of the search tree
--Considering adding the 1 feature
--Considering adding the 2 feature
--Considering adding the 3 feature
On level 2 i added feature 2 to current set
On the 3th level of the search tree
--Considering adding the 1 feature
--Considering adding the 3 feature
On level 3 i added feature 1 to current set
On the 4th level of the search tree
--Considering adding the 3 feature
On level 4 i added feature 3 to current set
```



For PART 1 of the Project#2 we completely divorce the **search part** from the **cross-validation part**.

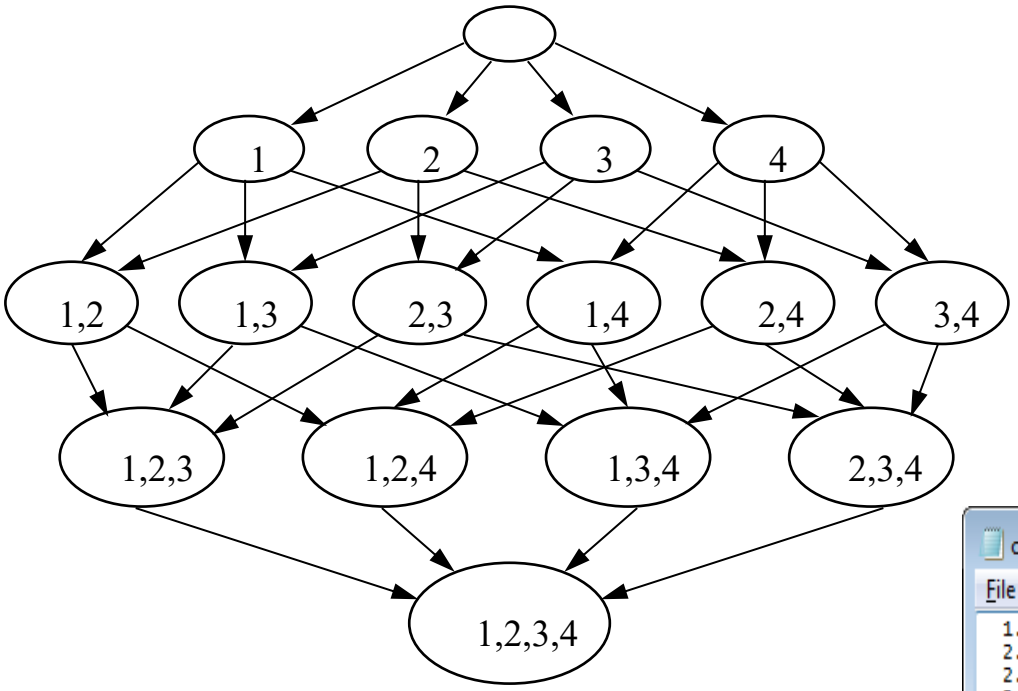
To do this, I use a stub function that just returns a random number

```
function accuracy= leave_one_out_cross_validation(data,current_set,feature_to_add)
    accuracy = rand;        % This is a testing stub only
end
```

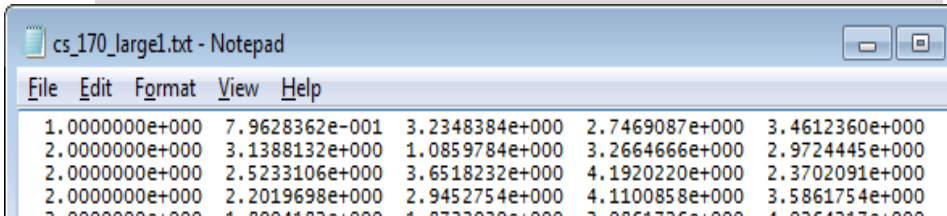
I will use this in my search algorithm, and only when I am 100% sure that search works, will I “fill in” the full leave-one-out-cross-validation code.

```
function  feature_search_demo(data)

  for i = 1 : size(data,2)-1
    disp(['On the ',num2str(i),'th level of the search tree'])
  end

end
```

I began by creating a **for** loop that can "walk" down the search tree.
I carefully tested it…

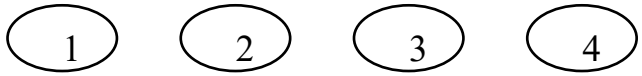



```
EDU>>
feature_search_demo(mydata)
On the 1th level of the search tree
On the 2th level of the search tree
On the 3th level of the search tree
On the 4th level of the search tree
```



```
function  feature_search_demo(data)

  for i = 1 : size(data,2)-1

    disp(['On the ',num2str(i),'th level of the search tree'])

    for k = 1 : size(data,2)-1

        disp(['--Considering adding the ', num2str(k),' feature'])

    end
  end
end
```

Now, inside the loop that "walks" down the search tree, I created a loop that considers each feature separately…
I carefully tested it…




```
EDU>> feature_search_demo(mydata)
On the 1th level of the search tree
--Considering adding the 1 feature
--Considering adding the 2 feature
--Considering adding the 3 feature
--Considering adding the 4 feature
On the 2th level of the search tree
--Considering adding the 1 feature
--Considering adding the 2 feature
--Considering adding the 3 feature
--Considering adding the 4 feature
On the 3th level of the search tree
--Considering adding the 1 feature
--Considering adding the 2 feature
--Considering adding the 3 feature
--Considering adding the 4 feature
On the 4th level of the search tree
--Considering adding the 1 feature
--Considering adding the 2 feature
--Considering adding the 3 feature
--Considering adding the 4 feature
```

```
function  feature_search_demo(data)

  for i = 1 : size(data,2)-1

    disp(['On the ',num2str(i),'th level of the search tree'])

    for k = 1 : size(data,2)-1

        disp(['--Considering adding the ', num2str(k),' feature'])

    end
  end
end
```

We are making great progress!

These nested loops are basically all we need to traverse the search space.

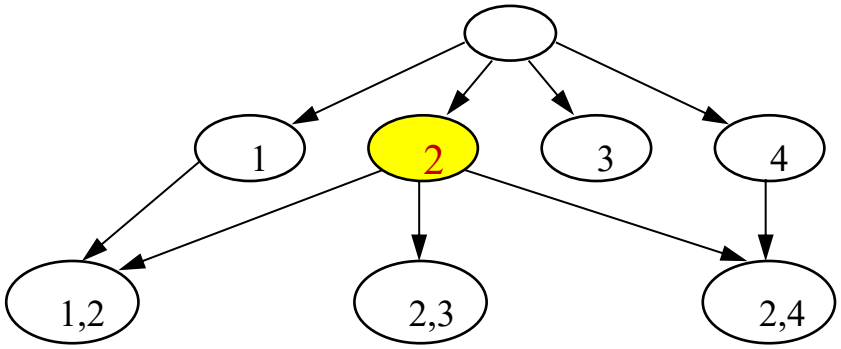
However at this point we are not measuring the accuracy of `leave_one_out_cross_validation` and recording it, so lets us do that (next slide).

The code below *almost* works, but, once you add a feature, you should not add it again…

```
feature_search_demo(mydata)
On the 1th level of the search tree
--Considering adding the 1 feature
--Considering adding the 2 feature
--Considering adding the 3 feature
--Considering adding the 4 feature
On level 1 i added feature 2 to current set
On the 2th level of the search tree
--Considering adding the 1 feature
--Considering adding the 2 feature
--Considering…
```

```
function  feature_search_demo(data)


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
        disp(['--Considering adding the ', num2str(k),' feature'])
        accuracy = leave_one_out_cross_validation(data,current_set_of_features,k);

        if accuracy > best_so_far_accuracy
            best_so_far_accuracy = accuracy;
            feature_to_add_at_this_level = k;
        end

    end

    disp(['On level ', num2str(i),' i added feature ', num2str(feature_to_add_at_this_level), ' to current set'])

 end
end
```

We need an IF statement in the inner loop that says "only consider adding this feature, if it was not already added" (next slide)



…We need an IF statement in the inner loop that says "only consider adding this feature, if it was not already added"

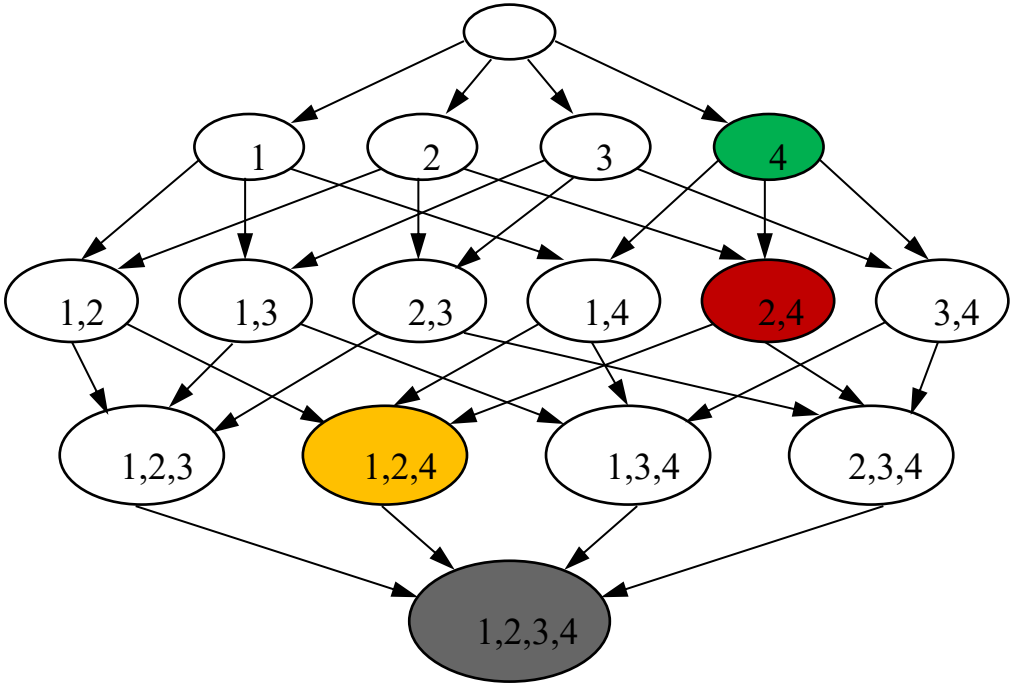
```
EDU>> feature_search_demo(mydata)
On the 1th level of the search tree
--Considering adding the 1 feature
--Considering adding the 2 feature
--Considering adding the 3 feature
--Considering adding the 4 feature
On level 1 i added feature 4 to current set
On the 2th level of the search tree
--Considering adding the 1 feature
--Considering adding the 2 feature
--Considering adding the 3 feature
On level 2 i added feature 2 to current set
On the 3th level of the search tree
--Considering adding the 1 feature
--Considering adding the 3 feature
On level 3 i added feature 1 to current set
On the 4th level of the search tree
--Considering adding the 3 feature
On level 4 i added feature 3 to current set
```

```
function  feature_search_demo(data)

current_set_of_features = []; % Initialize an empty set

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)) % Only consider adding, if not already added.
        disp(['--Considering adding the ', num2str(k),' feature'])
        accuracy = leave_one_out_cross_validation(data,current_set_of_features,k);

        if accuracy > best_so_far_accuracy
            best_so_far_accuracy = accuracy;
            feature_to_add_at_this_level = k;
        end
      end
     end

    current_set_of_features(i) =  feature_to_add_at_this_level;
    disp(['On level ', num2str(i),' i added feature ', num2str(feature_to_add_at_this_level), ' to current set'])

 end
end
```

We are done with the search!

The code is the previous slide is all you need. In part 2 you just have to replace the stub function `leave_one_out_cross_validation` with a real function

```
EDU>> feature_search_demo(mydata)
On the 1th level of the search tree
--Considering adding the 1 feature
--Considering adding the 2 feature
--Considering adding the 3 feature
--Considering adding the 4 feature
On level 1 i added feature 4 to current set
On the 2th level of the search tree
--Considering adding the 1 feature
--Considering adding the 2 feature
--Considering adding the 3 feature
On level 2 i added feature 2 to current set
On the 3th level of the search tree
--Considering adding the 1 feature
--Considering adding the 3 feature
On level 3 i added feature 1 to current set
On the 4th level of the search tree
--Considering adding the 3 feature
On level 4 i added feature 3 to current set
```



The second column up to the last column are the **features**

Class labels are in the first column Either a 1 or 2

cs_170_large1.txt - Notepad

File Edit Format View Help

1.0000000e+000	7.9628362e-001	3.2348384e+000	2.7469087e+000	3.4612360e+000
2.0000000e+000	3.1388132e+000	1.0859784e+000	3.2664666e+000	2.9724445e+000
2.0000000e+000	2.5233106e+000	3.6518232e+000	4.1920220e+000	2.3702091e+000
2.0000000e+000	2.2019698e+000	2.9452754e+000	4.1100858e+000	3.5861754e+000
2.0000000e+000	1.8904182e+000	1.8733939e+000	3.0861726e+000	4.0264217e+000
1.0000000e+000	2.8017969e+000	3.2018344e+000	2.4359749e+000	3.5219619e+000
2.0000000e+000	4.4191740e+000	2.6547288e+000	5.1146765e+000	4.0601259e+000
2.0000000e+000	2.9442884e+000	4.3319072e+000	2.7605042e+000	9.6102493e-001
2.0000000e+000	3.7413565e+000	2.9983523e+000	5.3396028e+000	3.0511236e+000
2.0000000e+000	2.2978020e+000	2.4119443e+000	3.2646000e+000	2.7864460e+000
2.0000000e+000	2.8229189e+000	4.3073637e+000	1.4185378e+000	2.0265755e+000
2.0000000e+000	4.3182128e+000	2.0439190e+000	3.7311071e+000	4.9233753e+000
2.0000000e+000	2.5799436e+000	3.0228907e+000	3.2035603e+000	2.0866863e+000
2.0000000e+000	2.1157746e+000	3.1532727e+000	1.8084600e+000	3.1541794e+000
2.0000000e+000	1.9756946e+000	4.8796698e+000	3.1833058e+000	4.4777842e+000
2.0000000e+000	3.4679338e+000	2.3722225e+000	1.2752609e+000	4.7518338e+000
2.0000000e+000	3.5901826e+000	2.9736468e+000	3.1323727e+000	2.1290448e+000
1.0000000e+000	3.4929911e+000	2.7731843e+000	2.2957511e+000	3.5733312e+000
1.0000000e+000	2.0263779e+000	4.6687710e+000	3.9057391e+000	3.2446302e+000
2.0000000e+000	3.7503586e+000	2.6710090e+000	2.0745432e+000	3.8080189e+000
2.0000000e+000	3.3174957e+000	1.9599153e+000	2.8730239e+000	2.0361295e+000
2.0000000e+000	3.7361733e+000	4.3694740e+000	1.5589263e+000	2.3920021e+000
2.0000000e+000	3.1377157e+000	9.9899049e-001	4.1298448e+000	2.6877226e+000
2.0000000e+000	4.2426458e+000	1.2532111e+000	2.4506267e+000	4.5661867e+000

These numbers are in standard IEEE 754-1985, single precision format (space delimited)

You can use an off-the-shelf package to read them into your program.

cs_170_large1.txt - Notepad

File Edit Format View Help

```
1.0000000e+000  7.9628362e-001  3.2348384e+000  2.7469087e+000  3.4612360e+000
2.0000000e+000  3.1388132e+000  1.0859784e+000  3.2664666e+000  2.9724445e+000
2.0000000e+000  2.5233106e+000  3.6518232e+000  4.1920220e+000  2.3702091e+000
2.0000000e+000  2.2019698e+000  2.9452754e+000  4.1100858e+000  3.5861754e+000
2.0000000e+000  1.8904182e+000  1.8733939e+000  3.0861726e+000  4.0264217e+000
1.0000000e+000  2.8017969e+000  3.2018344e+000  2.4359749e+000  3.5219619e+000
2.0000000e+000  4.4191740e+000  2.6547288e+000  5.1146765e+000  4.0601259e+000
2.0000000e+000  2.9442884e+000  4.3319072e+000  2.7605042e+000  9.6102493e-001
2.0000000e+000  3.7413565e+000  2.9983523e+000  5.3396028e+000  3.0511236e+000
2.0000000e+000  2.2978020e+000  2.4119443e+000  3.2646000e+000  2.7864460e+000
2.0000000e+000  2.8229189e+000  4.3073637e+000  1.4185378e+000  2.0265755e+000
2.0000000e+000  4.3182128e+000  2.0439190e+000  3.7311071e+000  4.9233753e+000
2.0000000e+000  2.5799436e+000  3.0228907e+000  3.2035603e+000  2.0866863e+000
2.0000000e+000  2.1157746e+000  3.1532727e+000  1.8084600e+000  3.1541794e+000
2.0000000e+000  1.9756946e+000  4.8796698e+000  3.1833058e+000  4.4777842e+000
2.0000000e+000  3.4679338e+000  2.3722225e+000  1.2752609e+000  4.7518338e+000
2.0000000e+000  3.5901826e+000  2.9736468e+000  3.1323727e+000  2.1290448e+000
1.0000000e+000  3.4929911e+000  2.7731843e+000  2.2957511e+000  3.5733312e+000
1.0000000e+000  2.0263779e+000  4.6687710e+000  3.9057391e+000  3.2446302e+000
2.0000000e+000  3.7503586e+000  2.6710090e+000  2.0745432e+000  3.8080189e+000
2.0000000e+000  3.3174957e+000  1.9599153e+000  2.8730239e+000  2.0361295e+000
2.0000000e+000  3.7361733e+000  4.3694740e+000  1.5589263e+000  2.3920021e+000
2.0000000e+000  3.1377157e+000  9.9899049e-001  4.1298448e+000  2.6877226e+000
2.0000000e+000  4.2426458e+000  1.2532111e+000  2.4506267e+000  4.5661867e+000
```

- I will now review the `leave_one_out_cross_validation` (PART 2 of your project)

- However, as you can see from these notes, you can work on the search and completely code it up now!

- I strongly recommend that you do so.