Project II briefing

Feature Selection with Nearest Neighbor

Due Date: Thursday of finals week at 11:59pm **No extension!**

The datasets will be online later this week, but you can start coding today.

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.

In this video, I will *only* discuss the search strategy
I will discus the classification part (cross validation) in a later video

Project II briefing

- Revisit slides on Nearest Neighbor Classification

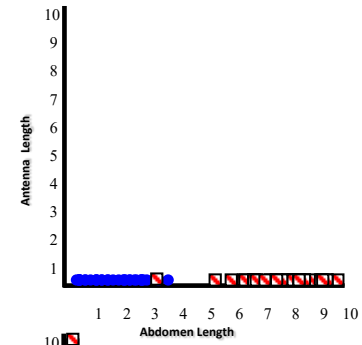
Recall our insect example…

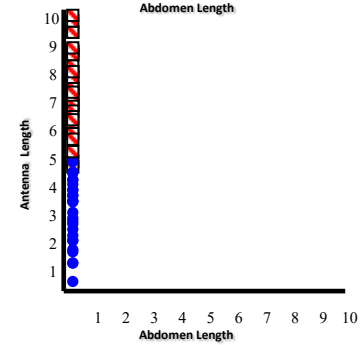
Suppose that we want to build a classifier for it.
We were given two features Abdomen length and Antenna length, but we don't have to use both.
We could use…

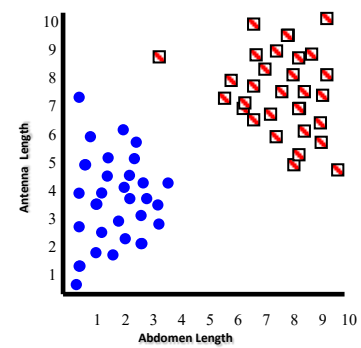
- Abdomen length only
- Antenna length only
- Both Abdomen length and Antenna length

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…

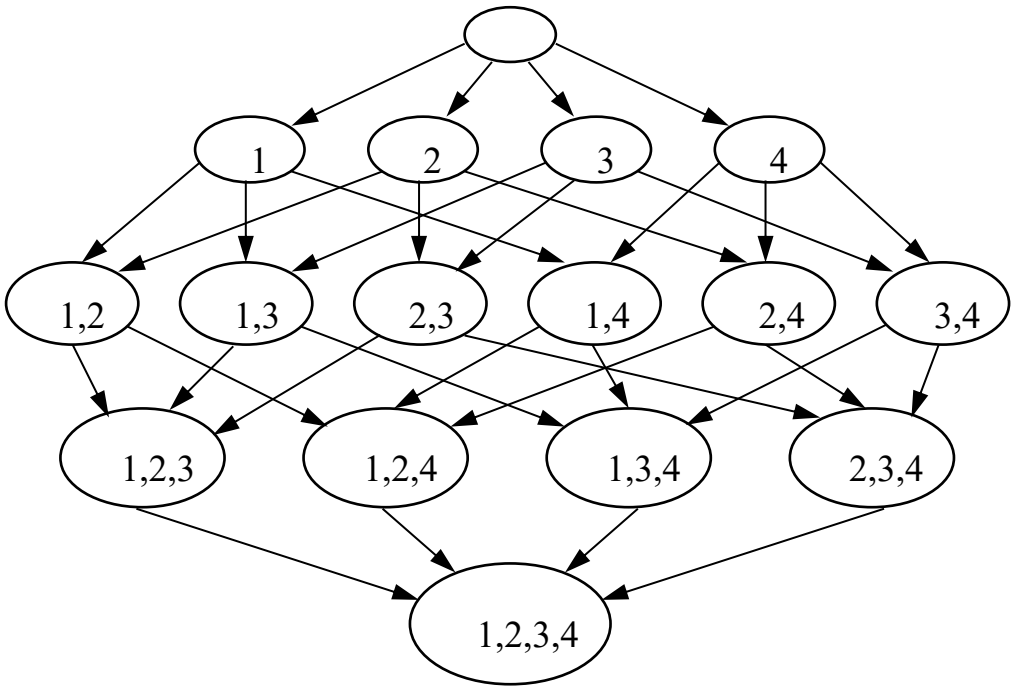
How can we pick the best features, when there are say 100s of them?







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.



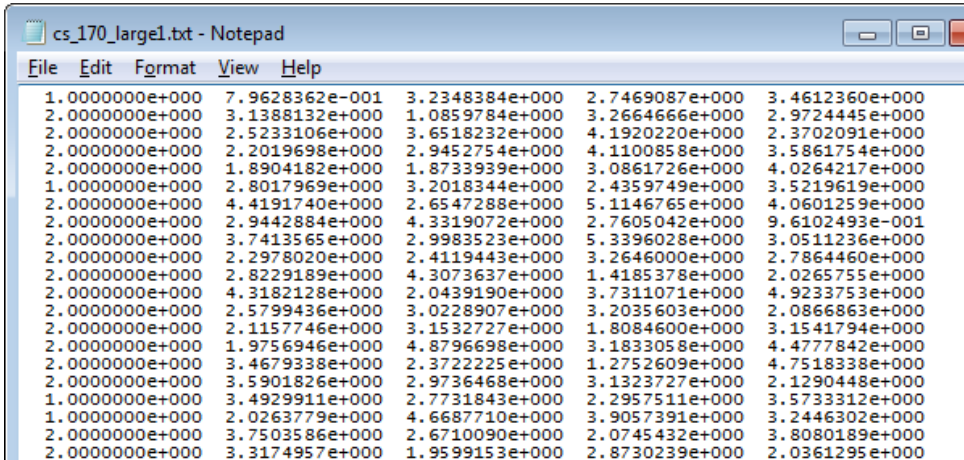
It might be that the problem is:

`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..

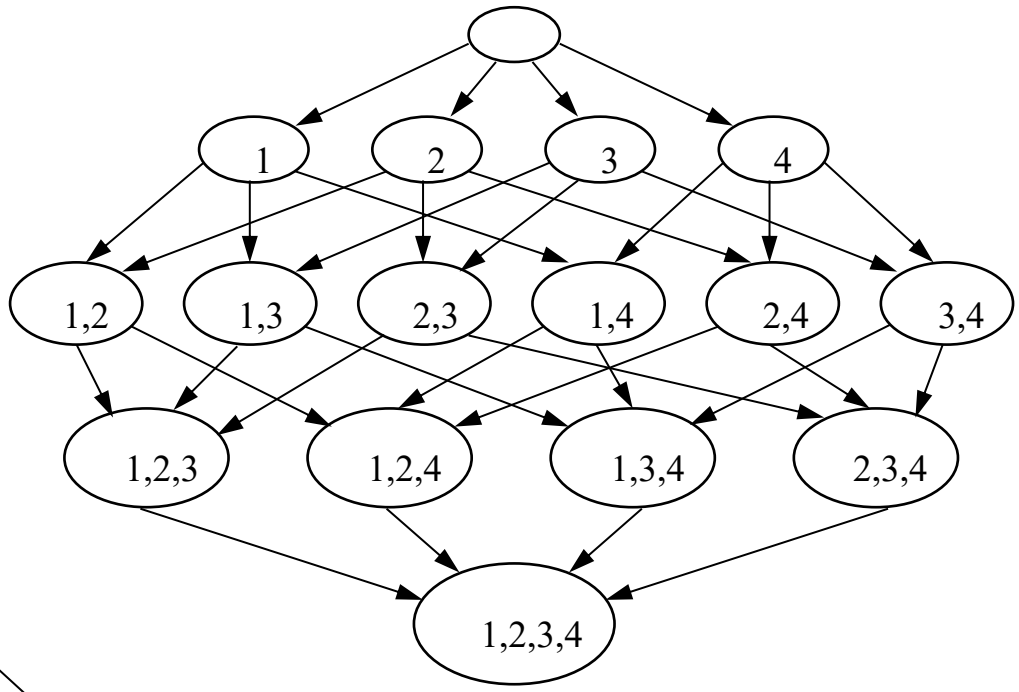


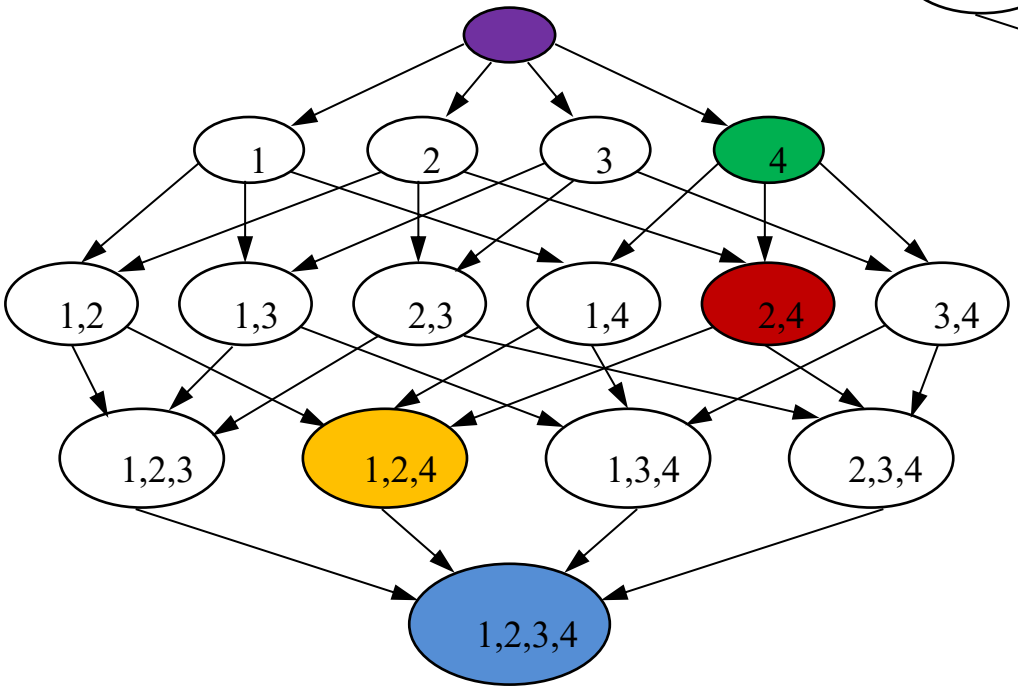
cs_170_large1.txt - Notepad

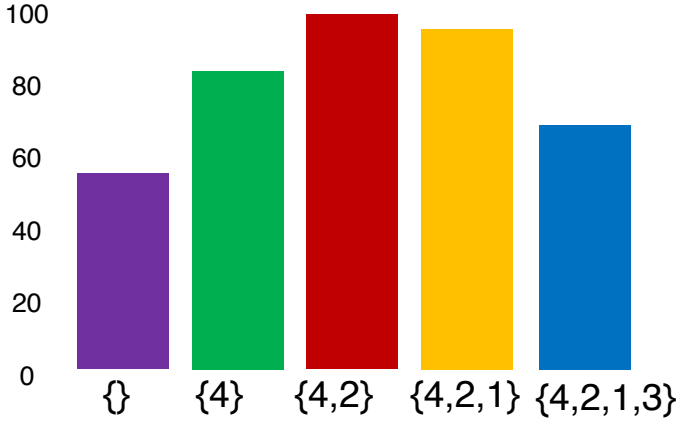
File Edit Format View Help

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

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.



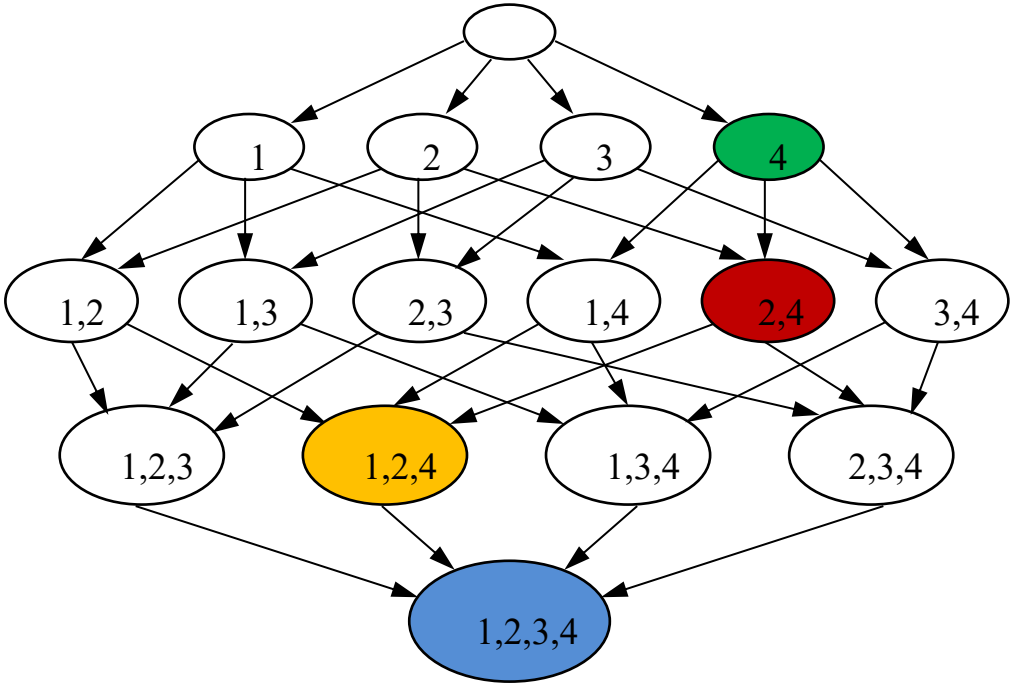




This is what project 2 search “looks” like. I just want the printout, the figure is for your ref only.

(I should have printed out the accuracy at each step, below you will see why I did not do that here)

```
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
```



I have a key for all the datasets.

For example, I know that

On large dataset 120 the accuracy rate can be 0.916 when using only features 91 79 95

In other words all the features are irrelevant, *except* for features 91 79 and 95

And I know that if you use ONLY those features, you can get an accuracy of about 0.916

You don’t have this key! So it is your job to do the search to find that subset of features.

Everyone will have a different subset and a different achievable accuracy

To finish this project, I recommend that you completely divorce the **search part**, from the **cross-validation part**.

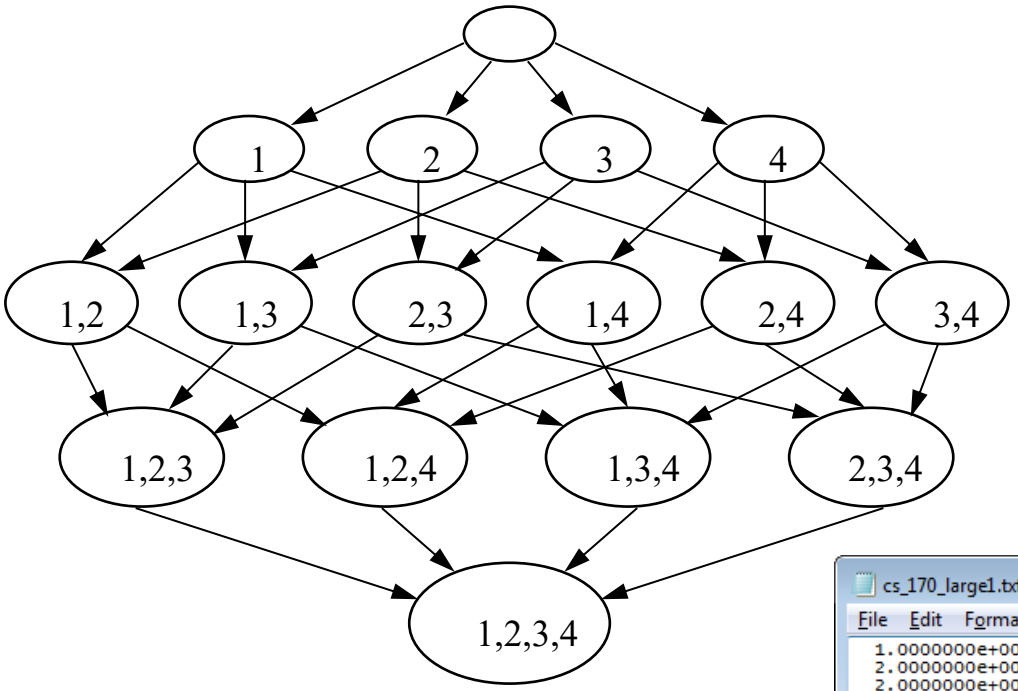
To do this, I wrote a stub function that just returns a random number

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

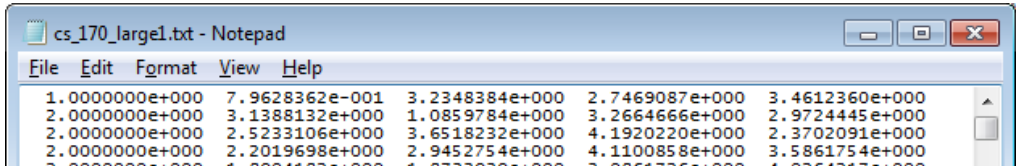
I will use this in my search algorithm, and only when I am 100% sure that search works, will I “fill in” the full leave-one-out-cross-validation code.

```
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…



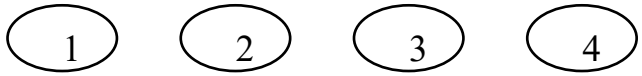

```
EDU>> feature_search_demo(mydata)
On the 1th level of the search tree
On the 2th level of the search tree
On the 3th level of the search tree
On the 4th level of the search tree
```



```
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.

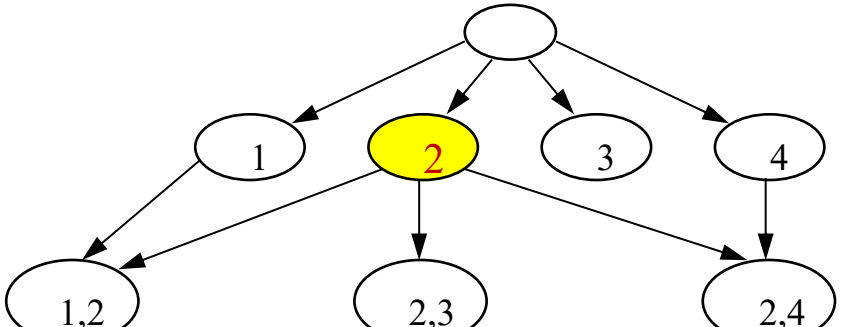
However at this point we are not measuring the accuracy of `leave_one_out_cross_validation` and recording it, so lets us do that (next slide).

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)

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
        disp(['--Considering adding the ', num2str(k),' feature'])
        accuracy = leave_one_out_cross_validation(data,current_set_of_features,k+1);

        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)



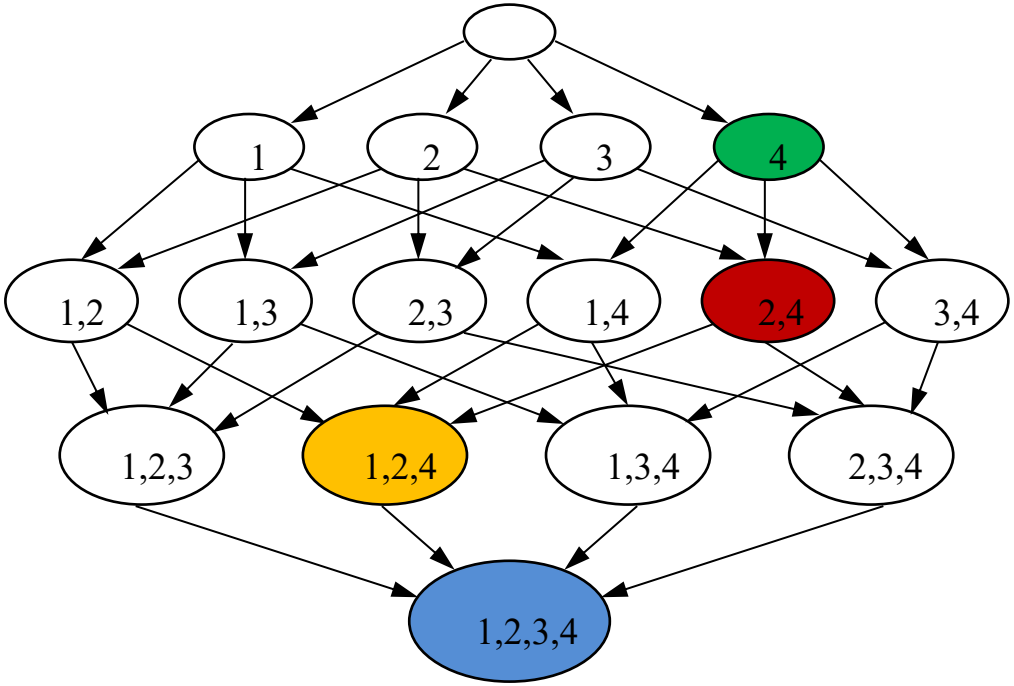
…We need an IF statement in the inner loop that says "only consider adding this feature, if it was not already added"

```
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+1);

         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.
You just have to replace the stub function `leave_one_out_cross_validation` with a real function, and echo the numbers it returned to the screen.




```
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
```

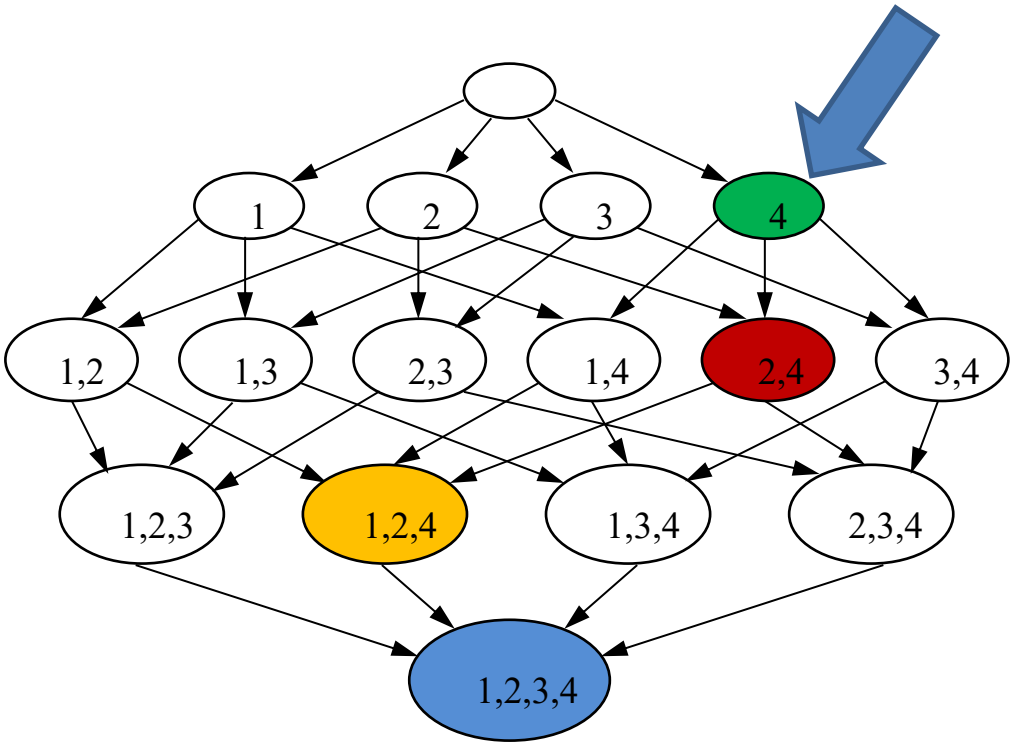
- I will review the `leave_one_out_cross_validation` part another time.
- However, as you can see from these notes, you can work on the search, and completely code it up now!
- I strongly recommend that you do so.

Part 2

What are we doing at each node?

We are running k-fold cross validation
This is a special case, where k is the number of objects in our dataset
(also called *leave-one-out*)
Depending on the node, we will be using various subsets of the features.

However, let us start by using *all* the features. I will write Matlab guide code..



Predictive Accuracy I

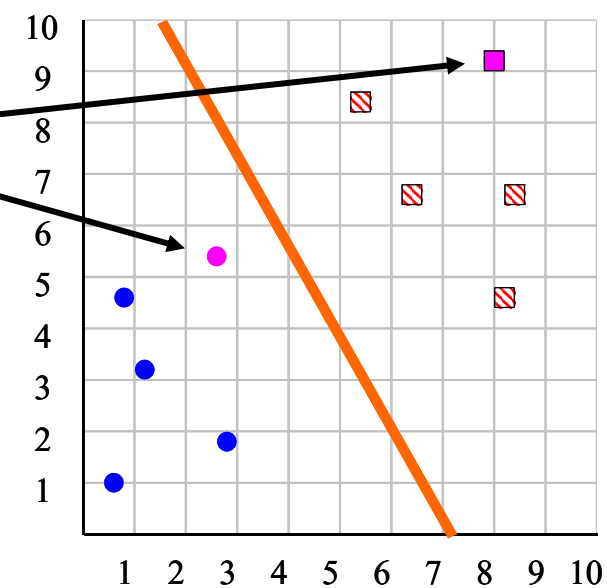
- How do we *estimate* the **accuracy** of our classifier?
  We can use ***K*-fold cross validation**

We divide the dataset into K equal sized sections. The algorithm is tested K times, each time leaving out one of the K section from building the classifier, but using it to *test* the classifier instead

$$\text{Accuracy} = \frac{\text{Number of correct classifications}}{\text{Number of instances in our database}}$$

$K = 5$

Insect ID	Abdomen Length	Antennae Length	Insect Class
1	2.7	5.5	Grasshopper
2	8.0	9.1	Katydid
3	0.9	4.7	Grasshopper
4	1.1	3.1	Grasshopper
5	5.4	8.5	Katydid
6	2.9	1.9	Grasshopper
7	6.1	6.6	Katydid
8	0.5	1.0	Grasshopper
9	8.3	6.6	Katydid
10	8.1	4.7	Katydids



I am going to test on this smaller dataset…

C:\Users\eamon\Documents\MATLAB\CS170_SMALLtestdata__1.txt

*CS170_SMALLtestdata__1.txt - Notepad

File Edit Format View Help

```
2.0000000e+00  -6.9166525e-01  -2.9439622e-01  -2.9222408e-01   8.7251996e-01   1.0483219e+00   1.7276280e+00   7.0041931e-01   2.6027058e-01  -1.2629121e+00  -4.5493399e-01
2.0000000e+00   3.5759969e-01   1.7038206e+00  -3.6101920e-01  -1.5651900e+00  -9.9701270e-02   1.1223806e+00   5.8018449e-01   1.6704111e-01  -3.5613544e-01  -1.5703011e+00
2.0000000e+00  -9.5816598e-01  -5.7519342e-01  -8.4971686e-02  -1.4798905e+00  -1.2459724e+00  -2.5840596e-01  -1.5131340e-01   4.9895775e-01  -9.7547158e-01  -2.2054287e-01
2.0000000e+00   4.9196756e-02   1.0812071e+00  -4.0229175e-01   1.1777236e+00   1.8673751e+00  -1.5164985e+00   2.8407496e-01  -2.4645325e-01   1.1343136e+00   1.9491297e+00
2.0000000e+00  -9.0648264e-01  -1.8374881e+00  -5.6314718e-01  -1.5433132e-01  -8.9188705e-01   1.2967436e+00   1.8286947e-01  -2.1527100e+00  -7.4133082e-01   2.0938863e-01
2.0000000e+00  -7.0580439e-01   1.3649122e-01  -1.0517689e-01   1.0847079e+00  -1.8650784e+00  -9.3006226e-01  -1.1301964e+00  -8.9560480e-01   7.6732439e-01   6.0920281e-01
1.0000000e+00  -3.0865657e-01   1.2043833e+00   1.0649033e+00   7.7815796e-01   7.8430174e-01  -3.8209179e-01   8.7954164e-01   6.1868397e-01   1.0864255e+00   3.6058458e-01
2.0000000e+00  -7.3322120e-01  -9.6302493e-01   7.1013829e-01  -8.4558380e-01   9.0538127e-01   8.4227111e-01   1.5129852e+00  -1.2740488e+00   1.0542377e+00  -4.2429762e-01
1.0000000e+00  -1.2792150e+00   1.0055919e+00   4.9778108e-01   1.0333189e-01  -8.7944164e-02  -4.8000544e-01   4.9205187e-01   3.7381626e-01  -1.0142017e+00  -2.3190544e+00
2.0000000e+00   2.4917676e+00  -2.2507010e-01  -7.1492203e-01   4.8350054e-03   1.5717230e-01   2.57484S61e-01   7.7375728e-02  -2.0335414e+00   1.8954952e-01  -1.8521299e+00
```

Ln 10, Col 108 | 100% | Windows (CRLF) | UTF-8

K = the number of rows (here 10)

$K = 5$

Insect ID	Abdomen Length	Antennae Length	Insect Class
1	2.7	5.5	**Grasshopper**
2	8.0	9.1	**Katydid**
3	0.9	4.7	**Grasshopper**
4	1.1	3.1	**Grasshopper**
5	5.4	8.5	**Katydid**
6	2.9	1.9	**Grasshopper**
7	6.1	6.6	**Katydid**
8	0.5	1.0	**Grasshopper**
9	8.3	6.6	**Katydid**
10	8.1	4.7	**Katydids**

```
function accuracy = cs170demo()
data = load('C:\Users\eamon\Documents\MATLAB\CS170_SMALLtestdata__1.txt');

 for i = 1 : size(data,1)
    object_to_classify = data(i,2:end);
    label_object_to_classify = data(i,1);


    disp(['Looping over i, at the ',int2str(i),' location']);
    disp(['The ',int2str(i),'th object is in class ',num2str(label_object_to_classify)]);


 end
end
```

```
Looping over i, at the 1 location
The 1th object is in class 2
Looping over i, at the 2 location
The 2th object is in class 2
Looping over i, at the 3 location
The 3th object is in class 2
Looping over i, at the 4 location
The 4th object is in class 2
Looping over i, at the 5 location
The 5th object is in class 2
Looping over i, at the 6 location
The 6th object is in class 2
Looping over i, at the 7 location
The 7th object is in class 1
Looping over i, at the 8 location
The 8th object is in class 2
Looping over i, at the 9 location
The 9th object is in class 1
Looping over i, at the 10 location
The 10th object is in class 2
```

*CS170_SMALLtestdata__1.txt - Notepad

File Edit Format View Help

2.0000000e+00	-6.9166525e-01	-2.9439622e-01	-2.9222408e-01	8.7251996e-01	1.0483219e+00	1.72
2.0000000e+00	3.5759969e-01	1.7038206e+00	-3.6101920e-01	-1.5651900e+00	-9.9701270e-02	1.12
2.0000000e+00	-9.5816598e-01	-5.7519342e-01	-8.4971686e-02	-1.4798905e+00	-1.2459724e+00	-2.58
2.0000000e+00	4.9196756e-02	1.0812071e+00	-4.0229175e-01	1.1777236e+00	1.8673751e+00	-1.51
2.0000000e+00	-9.0648264e-01	-1.8374881e+00	-5.6314718e-01	-1.5433132e-01	-8.9188705e-01	1.29
2.0000000e+00	-7.0580439e-01	1.3649122e-01	-1.0517689e-01	1.0847079e+00	-1.8650784e+00	-9.30
1.0000000e+00	-3.0865657e-01	1.2043833e+00	1.0649033e+00	7.7815796e-01	7.8430174e-01	-3.82
2.0000000e+00	-7.3322120e-01	-9.6302493e-01	7.1013829e-01	-8.4558380e-01	9.0538127e-01	8.42
1.0000000e+00	-1.2792150e+00	1.0055919e+00	4.9778108e-01	1.0333189e-01	-8.7944164e-02	-4.80
2.0000000e+00	2.4917676e+00	-2.2507010e-01	-7.1492203e-01	4.8350054e-03	1.5717230e-01	2.57

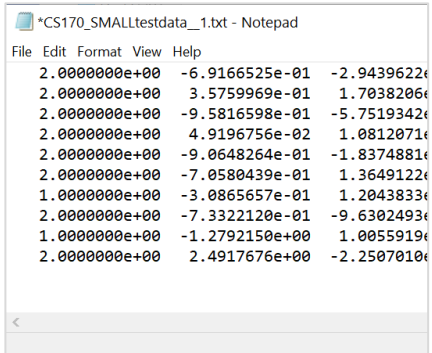
```
function accuracy = cs170demo()
data = load('C:\Users\eamon\Documents\MATLAB\CS170_SMALLtestdata__1.txt');

 for i = 1 : size(data,1)
    object_to_classify = data(i,2:end);
    label_object_to_classify = data(i,1);

    nearest_neighbor_distance = inf;
    nearest_neighbor_location = inf;
    for k = 1 : size(data,1)
        disp(['Ask if ',int2str(i),' is nearest neigbour with ', int2str(k)])

    end

 end
end
```

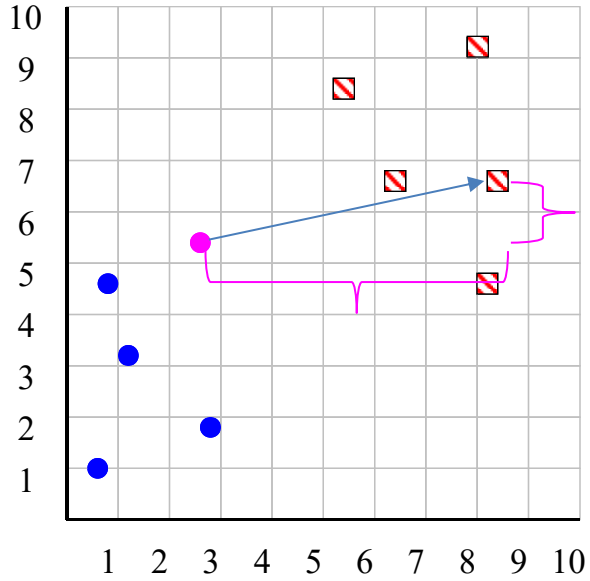
```
Ask if 1 is nearest neigbour with 1
Ask if 1 is nearest neigbour with 2
Ask if 1 is nearest neigbour with 3
Ask if 1 is nearest neigbour with 4
Ask if 1 is nearest neigbour with 5
Ask if 1 is nearest neigbour with 6
Ask if 1 is nearest neigbour with 7
Ask if 1 is nearest neigbour with 8
Ask if 1 is nearest neigbour with 9
Ask if 1 is nearest neigbour with 10
Ask if 2 is nearest neigbour with 1
Ask if 2 is nearest neigbour with 2
Ask if 2 is nearest neigbour with 3
Ask if 2 is nearest neigbour with 4
Ask if 2 is nearest neigbour with 5
Ask if 2 is nearest neigbour with 6
Ask if 2 is nearest neigbour with 7
Ask if 2 is nearest neigbour with 8
Ask if 2 is nearest neigbour with 9
Ask if 2 is nearest neigbour with 10
Ask if 3 is nearest neigbour with 1
Ask if 3 is nearest neigbour with 2
Ask if 3 is nearest neigbour with 3
...
```

```
function accuracy = cs170demo()
data = load('C:\Users\eamon\Documents\MATLAB\CS170_SMALLtestdata__1.txt');

 for i = 1 : size(data,1)
    object_to_classify = data(i,2:end);
    label_object_to_classify = data(i,1);

    nearest_neighbor_distance = inf;
    nearest_neighbor_location = inf;
    for k = 1 : size(data,1)


        if k ~= i  % don't compare to yourself!!!
           disp(['Ask if ',int2str(i),' is nearest neigbour with ', int2str(k)])
        end
    end

 end
end
```

```
Ask if 1 is nearest neigbour with 2
Ask if 1 is nearest neigbour with 3
Ask if 1 is nearest neigbour with 4
Ask if 1 is nearest neigbour with 5
Ask if 1 is nearest neigbour with 6
Ask if 1 is nearest neigbour with 7
Ask if 1 is nearest neigbour with 8
Ask if 1 is nearest neigbour with 9
Ask if 1 is nearest neigbour with 10
Ask if 2 is nearest neigbour with 1
Ask if 2 is nearest neigbour with 3
Ask if 2 is nearest neigbour with 4
Ask if 2 is nearest neigbour with 5
Ask if 2 is nearest neigbour with 6
Ask if 2 is nearest neigbour with 7
Ask if 2 is nearest neigbour with 8
Ask if 2 is nearest neigbour with 9
Ask if 2 is nearest neigbour with 10
Ask if 3 is nearest neigbour with 1
Ask if 3 is nearest neigbour with 2
Ask if 3 is nearest neigbour with 4
```

```
function accuracy = cs170demo()
data = load('C:\Users\eamon\Documents\MATLAB\CS170_SMALLtestdata__1.txt');

 for i = 1 : size(data,1)
    object_to_classify = data(i,2:end);
    label_object_to_classify = data(i,1);

    nearest_neighbor_distance = inf;
    nearest_neighbor_location = inf;
    for k = 1 : size(data,1)
        disp(['Ask if ',int2str(i),' is nearest neigbour with ', int2str(k)])

         if k ~= i
              distance = sqrt(sum((object_to_classify - data(k,2:end)).^2));
              if  distance <   nearest_neighbor_distance
                     nearest_neighbor_distance = distance;
                     nearest_neighbor_location = k;
                     nearest_neighbor_label    = data(nearest_neighbor_location,1);
              end
         end
    end

 end
end
```





```
function accuracy = cs170demo()
data = load('C:\Users\eamon\Documents\MATLAB\CS170_SMALLtestdata__1.txt');

 for i = 1 : size(data,1)
    object_to_classify = data(i,2:end);
    label_object_to_classify = data(i,1);

    nearest_neighbor_distance = inf;
    nearest_neighbor_location = inf;
    for k = 1 : size(data,1)
        if k ~= i
             distance = sqrt(sum((object_to_classify - data(k,2:end)).^2));
             if  distance <   nearest_neighbor_distance
                    nearest_neighbor_distance = distance;
                    nearest_neighbor_location = k;
                    nearest_neighbor_label    = data(nearest_neighbor_location,1);
             end
        end
    end

    disp(['Object ', num2str(i), ' is class ', num2str(label_object_to_classify)]);
    disp(['Its nearest_neighbor is ', num2str(nearest_neighbor_location), ' which is in class ', num2str( nearest_neighbor_label )]);

 end
end
```

```
Object 1 is class 2
Its nearest_neighbor is 9 which is in class 1
Object 2 is class 2
Its nearest_neighbor is 9 which is in class 1
Object 3 is class 2
Its nearest_neighbor is 9 which is in class 1
Object 4 is class 2
Its nearest_neighbor is 7 which is in class 1
Object 5 is class 2
Its nearest_neighbor is 8 which is in class 2
```

*CS170_SMALLtestdata__1.txt - Notepad

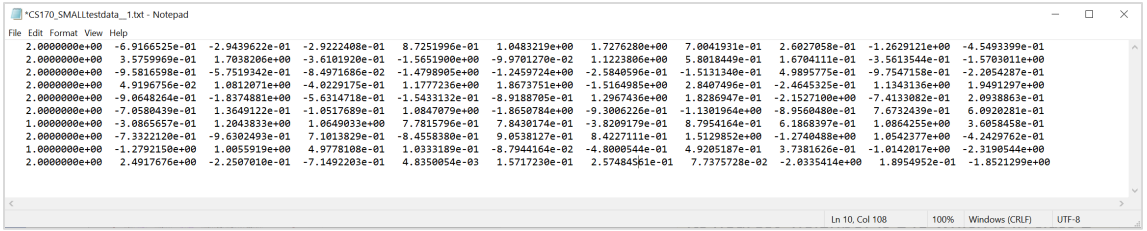
File Edit Format View Help

```
2.0000000e+00  -6.9166525e-01  -2.9439622e
2.0000000e+00   3.5759969e-01   1.7038206e
2.0000000e+00  -9.5816598e-01  -5.7519342e
2.0000000e+00   4.9196756e-02   1.0812071e
2.0000000e+00  -9.0648264e-01  -1.8374881e
2.0000000e+00  -7.0580439e-01   1.3649122e
1.0000000e+00  -3.0865657e-01   1.2043833e
2.0000000e+00  -7.3322120e-01  -9.6302493e
1.0000000e+00  -1.2792150e+00   1.0055919e
2.0000000e+00   2.4917676e+00  -2.2507010e
```

```
function accuracy = cs170demo()
data = load('C:\Users\eamon\Documents\MATLAB\CS170_SMALLtestdata__1.txt');
number_correctly_classfied = 0;

 for i = 1 : size(data,1)
    object_to_classify = data(i,2:end);
    label_object_to_classify = data(i,1);

    nearest_neighbor_distance = inf;
    nearest_neighbor_location = inf;
    for k = 1 : size(data,1)
        if k ~= i
             distance = sqrt(sum((object_to_classify - data(k,2:end)).^2));
             if  distance <   nearest_neighbor_distance
                    nearest_neighbor_distance = distance;
                    nearest_neighbor_location = k;
                    nearest_neighbor_label    = data(nearest_neighbor_location,1);
             end
        end
    end

    if label_object_to_classify == nearest_neighbor_label;
         number_correctly_classfied = number_correctly_classfied + 1;
    end

 end
 accuracy = number_correctly_classfied / size(data,1);
end
```

The code above is 90% of what you need!!!

This function returns 0.4000

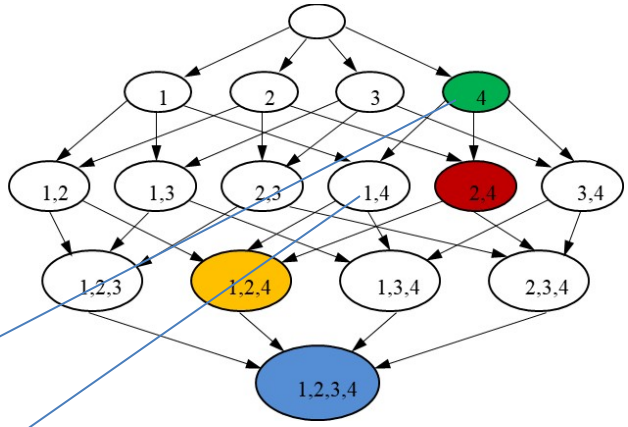


*CS170_SMALLtestdata__1.txt - Notepad

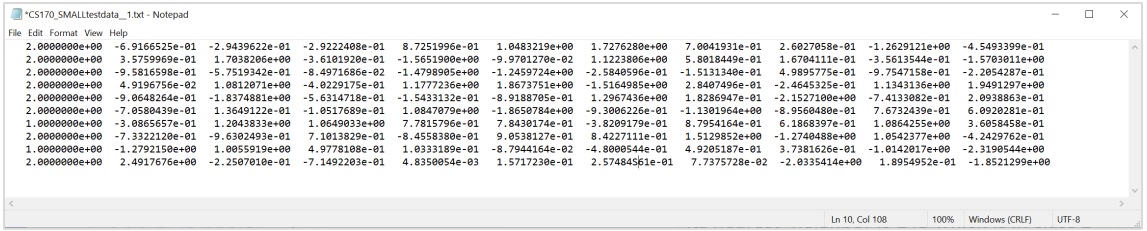
File Edit Format View Help

2.0000000e+00	-6.9166525e-01	-2.9439622e-01	-2.9222408e-01	8.7251996e-01	1.0483219e+00	1.7276280e+00	7.0041931e-01	2.6027058e-01	-1.2629121e+00	-4.5493399e-01
2.0000000e+00	3.5759969e-01	1.7038206e+00	-3.6101920e-01	-1.5651900e+00	-9.9701270e-02	1.1223806e+00	5.8018449e-01	1.6704111e-01	-3.5613544e-01	-1.5703011e+00
2.0000000e+00	-9.5816598e-01	-5.7519342e-01	-8.4971686e-02	-1.4798905e+00	-1.2459724e+00	-2.5840596e-01	-1.5131340e-01	4.9895775e-01	-9.7547158e-01	-2.2054287e-01
2.0000000e+00	4.9196756e-02	1.0812071e+00	-4.0229175e-01	1.1777236e+00	1.8673751e+00	-1.5164985e+00	2.8407496e-01	-2.4645325e-01	1.1343136e+00	1.9491297e+00
2.0000000e+00	-9.0648264e-01	-1.8374881e+00	-5.6314718e-01	-1.5433132e-01	-8.9188705e-01	1.2967436e+00	1.8286947e-01	-2.1527100e+00	-7.4133082e-01	2.0938863e-01
2.0000000e+00	-7.0580439e-01	1.3649122e-01	-1.0517689e-01	1.0847079e+00	-1.8650784e+00	-9.3006226e-01	-1.1301964e+00	-8.9560480e-01	7.6732439e-01	6.0920281e-01
1.0000000e+00	-3.0865657e-01	1.2043833e+00	1.0649033e+00	7.7815796e-01	7.8430174e-01	-3.8209179e-01	8.7954164e-01	6.1868397e-01	1.0864255e+00	3.6058458e-01
2.0000000e+00	-7.3322120e-01	-9.6302493e-01	7.1013829e-01	-8.4558380e-01	9.0538127e-01	8.4227111e-01	1.5129852e+00	-1.2740488e+00	1.0542377e+00	-4.2429762e-01
1.0000000e+00	-1.2792150e+00	1.0055919e+00	4.9778108e-01	1.0333189e-01	-8.7944164e-02	-4.8000544e-01	4.9205187e-01	3.7381626e-01	-1.0142017e+00	-2.3190544e+00
2.0000000e+00	2.4917676e+00	-2.2507010e-01	-7.1492203e-01	4.8350054e-03	1.5717230e-01	2.5748456e-01	7.7375728e-02	-2.0335414e+00	1.8954952e-01	-1.8521299e+00

Ln 10, Col 108 100% Windows (CRLF) UTF-8




```
function accuracy= leave_one_out_cross_validation(data,current_set,feature_to_add)

number_correctly_classfied = 0;

 for i = 1 : size(data,1)
    object_to_classify = data(i,2:end);
    label_object_to_classify = data(i,1);

    nearest_neighbor_distance = inf;
    nearest_neighbor_location = inf;
    for k = 1 : size(data,1)
        if k ~= i
             distance = sqrt(sum((object_to_classify - data(k,2:end)).^2));
             if  distance <   nearest_neighbor_distance
                    nearest_neighbor_distance = distance;
                    nearest_neighbor_location = k;
                    nearest_neighbor_label    = data(nearest_neighbor_location,1);
             end
        end
    end

    if label_object_to_classify == nearest_neighbor_label;
         number_correctly_classfied = number_correctly_classfied + 1;
    end

 end
 accuracy = number_correctly_classfied / size(data,1);
end
```



*CS170_SMALLtestdata__1.txt - Notepad

File Edit Format View Help

2.0000000e+00	-6.9166525e-01	-2.9439622e-01	-2.9222408e-01	8.7251996e-01	1.0483219e+00	1.7276280e+00	7.0041931e-01	2.6027058e-01	-1.2629121e+00	-4.5493399e-01
2.0000000e+00	3.5759969e-01	1.7038206e+00	-3.6101920e-01	-1.5651900e+00	-9.9701270e-02	1.1223806e+00	5.8018449e-01	1.6704111e-01	-3.5613544e-01	-1.5703011e+00
2.0000000e+00	-9.5816598e-01	-5.7519342e-01	-8.4971686e-02	-1.4798905e+00	-1.2459724e+00	-2.5840596e-01	-1.5131340e-01	4.9895775e-01	-9.7547158e-01	-2.2054287e-01
2.0000000e+00	4.9196756e-02	1.0812071e+00	-4.0229175e-01	1.1777236e+00	1.8673751e+00	-1.5164985e+00	2.8407496e-01	-2.4645325e-01	1.1343136e+00	1.9491297e+00
2.0000000e+00	-9.0648264e-01	-1.8374881e+00	-5.6314718e-01	-1.5433132e-01	-8.9188705e-01	1.2967436e+00	1.8286947e-01	-2.1527100e+00	-7.4133082e-01	2.0938863e-01
2.0000000e+00	-7.0580439e-01	1.3649122e-01	-1.0517689e-01	1.0847079e+00	-1.8650784e+00	-9.3006226e-01	-1.1301964e+00	-8.9560480e-01	7.6732439e-01	6.0920281e-01
1.0000000e+00	-3.0865657e-01	1.2043833e+00	1.0649033e+00	7.7815796e-01	7.8430174e-01	-3.8209179e-01	8.7954164e-01	6.1868397e-01	1.0864255e+00	3.6058458e-01
2.0000000e+00	-7.3322120e-01	-9.6302493e-01	7.1013829e-01	-8.4558380e-01	9.0538127e-01	8.4227111e-01	1.5129852e+00	-1.2740488e+00	1.0542377e+00	-4.2429762e-01
1.0000000e+00	-1.2792150e+00	1.0055919e+00	4.9778108e-01	1.0333189e-01	-8.7944164e-02	-4.8000544e-01	4.9205187e-01	3.7381626e-01	-1.0142017e+00	-2.3190544e+00
2.0000000e+00	2.4917676e+00	-2.2507010e-01	-7.1492203e-01	4.8350054e-03	1.5717230e-01	2.5748456e-01	7.7375728e-02	-2.0335414e+00	1.8954952e-01	-1.8521299e+00

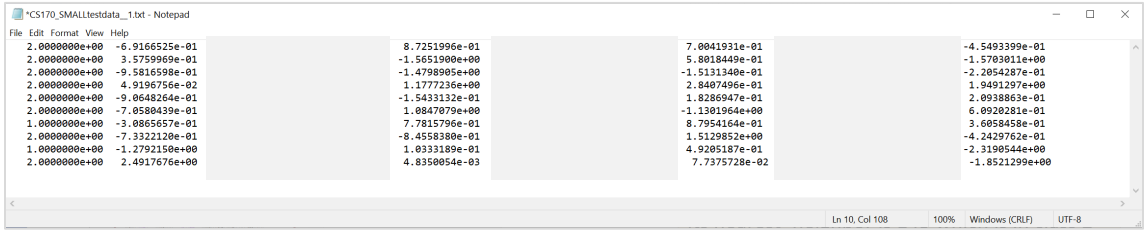
Ln 10, Col 108 100% Windows (CRLF) UTF-8

1 4 7

10

```
function accuracy= leave_one_out_cross_validation(data,current_set,feature_to_add)
```

Some code to do this…



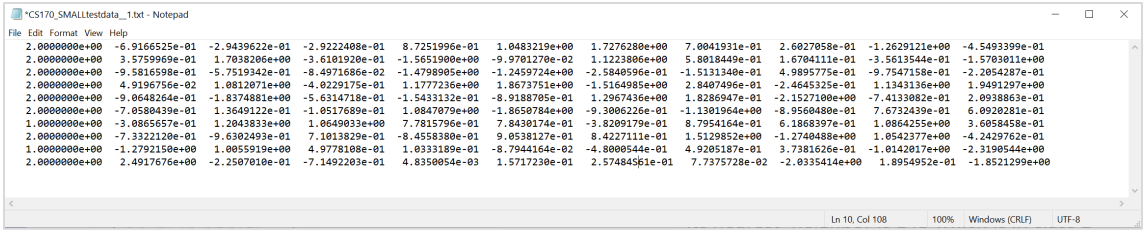
*CS170_SMALLtestdata__1.txt - Notepad

File Edit Format View Help

2.0000000e+00	-6.9166525e-01	8.7251996e-01	7.0041931e-01	-4.5493399e-01
2.0000000e+00	3.5759969e-01	-1.5651900e+00	5.8018449e-01	-1.5703011e+00
2.0000000e+00	-9.5816598e-01	-1.4798905e+00	-1.5131340e-01	-2.2054287e-01
2.0000000e+00	4.9196756e-02	1.1777236e+00	2.8407496e-01	1.9491297e+00
2.0000000e+00	-9.0648264e-01	-1.5433132e-01	1.8286947e-01	2.0938863e-01
2.0000000e+00	-7.0580439e-01	1.0847079e+00	-1.1301964e+00	6.0920281e-01
1.0000000e+00	-3.0865657e-01	7.7815796e-01	8.7954164e-01	3.6058458e-01
2.0000000e+00	-7.3322120e-01	-8.4558380e-01	1.5129852e+00	-4.2429762e-01
1.0000000e+00	-1.2792150e+00	1.0333189e-01	4.9205187e-01	-2.3190544e+00
2.0000000e+00	2.4917676e+00	4.8350054e-03	7.7375728e-02	-1.8521299e+00

Ln 10, Col 108 100% Windows (CRLF) UTF-8

```
number_correctly_classfied = 0;

 for i = 1 : size(data,1)
    object_to_classify = data(i,2:end);
    label_object_to_classify = data(i,1);

    nearest_neighbor_distance = inf;
    nearest_neighbor_location = inf;
    for k = 1 : size(data,1)
        if k ~= i
             distance = sqrt(sum((object_to_classify - data(k,2:end)).^2));
             if  distance <   nearest_neighbor_distance
                    nearest_neighbor_distance = distance;
                    nearest_neighbor_location = k;
                    nearest_neighbor_label    = data(nearest_neighbor_location,1);
             end
        end
    end

    if label_object_to_classify == nearest_neighbor_label;
         number_correctly_classfied = number_correctly_classfied + 1;
    end

 end
 accuracy = number_correctly_classfied / size(data,1);
end
```



*CS170_SMALLtestdata__1.txt - Notepad

File Edit Format View Help

2.0000000e+00	-6.9166525e-01	-2.9439622e-01	-2.9222408e-01	8.7251996e-01	1.0483219e+00	1.7276280e+00	7.0041931e-01	2.6027058e-01	-1.2629121e+00	-4.5493399e-01
2.0000000e+00	3.5759969e-01	1.7038206e+00	-3.6101920e-01	-1.5651900e+00	-9.9701270e-02	1.1223806e+00	5.8018449e-01	1.6704111e-01	-3.5613544e-01	-1.5703011e+00
2.0000000e+00	-9.5816598e-01	-5.7519342e-01	-8.4971686e-02	-1.4798905e+00	-1.2459724e+00	-2.5840596e-01	-1.5131340e-01	4.9895775e-01	-9.7547158e-01	-2.2054287e-01
2.0000000e+00	4.9196756e-02	1.0812071e+00	-4.0229175e-01	1.1777236e+00	1.8673751e+00	-1.5164985e+00	2.8407496e-01	-2.4645325e-01	1.1343136e+00	1.9491297e+00
2.0000000e+00	-9.0648264e-01	-1.8374881e+00	-5.6314718e-01	-1.5433132e-01	-8.9188705e-01	1.2967436e+00	1.8286947e-01	-2.1527100e+00	-7.4133082e-01	2.0938863e-01
2.0000000e+00	-7.0580439e-01	1.3649122e-01	-1.0517689e-01	1.0847079e+00	-1.8650784e+00	-9.3006226e-01	-1.1301964e+00	-8.9560480e-01	7.6732439e-01	6.0920281e-01
1.0000000e+00	-3.0865657e-01	1.2043833e+00	1.0649033e+00	7.7815796e-01	7.8430174e-01	-3.8209179e-01	8.7954164e-01	6.1868397e-01	1.0864255e+00	3.6058458e-01
2.0000000e+00	-7.3322120e-01	-9.6302493e-01	7.1013829e-01	-8.4558380e-01	9.0538127e-01	8.4227111e-01	1.5129852e+00	-1.2740488e+00	1.0542377e+00	-4.2429762e-01
1.0000000e+00	-1.2792150e+00	1.0055919e+00	4.9778108e-01	1.0333189e-01	-8.7944164e-02	-4.8000544e-01	4.9205187e-01	3.7381626e-01	-1.0142017e+00	-2.3190544e+00
2.0000000e+00	2.4917676e+00	-2.2507010e-01	-7.1492203e-01	4.8350054e-03	1.5717230e-01	2.57484$61e-01	7.7375728e-02	-2.0335414e+00	1.8954952e-01	-1.8521299e+00

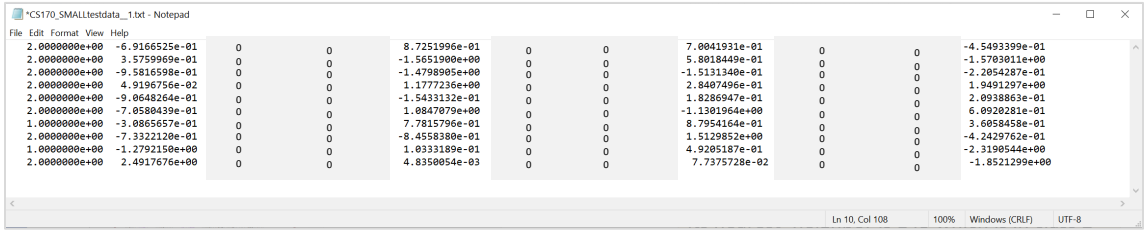
Ln 10, Col 108 100% Windows (CRLF) UTF-8

1, 4, 7

10

```
function accuracy= leave_one_out_cross_validation(data,current_set,feature_to_add)
```

Some code to do this…



*CS170_SMALLtestdata__1.txt - Notepad

File Edit Format View Help

2.0000000e+00	-6.9166525e-01	0	0	8.7251996e-01	0	0	7.0041931e-01	0	0	-4.5493399e-01
2.0000000e+00	3.5759969e-01	0	0	-1.5651900e+00	0	0	5.8018449e-01	0	0	-1.5703011e+00
2.0000000e+00	-9.5816598e-01	0	0	-1.4798905e+00	0	0	-1.5131340e-01	0	0	-2.2054287e-01
2.0000000e+00	4.9196756e-02	0	0	1.1777236e+00	0	0	2.8407496e-01	0	0	1.9491297e+00
2.0000000e+00	-9.0648264e-01	0	0	-1.5433132e-01	0	0	1.8286947e-01	0	0	2.0938863e-01
2.0000000e+00	-7.0580439e-01	0	0	1.0847079e+00	0	0	-1.1301964e+00	0	0	6.0920281e-01
1.0000000e+00	-3.0865657e-01	0	0	7.7815796e-01	0	0	8.7954164e-01	0	0	3.6058458e-01
2.0000000e+00	-7.3322120e-01	0	0	-8.4558380e-01	0	0	1.5129852e+00	0	0	-4.2429762e-01
1.0000000e+00	-1.2792150e+00	0	0	1.0333189e-01	0	0	4.9205187e-01	0	0	-2.3190544e+00
2.0000000e+00	2.4917676e+00	0	0	4.8350054e-03	0	0	7.7375728e-02	0	0	-1.8521299e+00

Ln 10, Col 108 100% Windows (CRLF) UTF-8

```
number_correctly_classfied = 0;

 for i = 1 : size(data,1)
    object_to_classify = data(i,2:end);
    label_object_to_classify = data(i,1);

    nearest_neighbor_distance = inf;
    nearest_neighbor_location = inf;
    for k = 1 : size(data,1)
        if k ~= i
             distance = sqrt(sum((object_to_classify - data(k,2:end)).^2));
             if  distance <   nearest_neighbor_distance
                    nearest_neighbor_distance = distance;
                    nearest_neighbor_location = k;
                    nearest_neighbor_label    = data(nearest_neighbor_location,1);
             end
        end
    end

    if label_object_to_classify == nearest_neighbor_label;
         number_correctly_classfied = number_correctly_classfied + 1;
    end

 end
 accuracy = number_correctly_classfied / size(data,1);
end
```

Done!!!!