Project II briefing

Feature Selection with Nearest Neighbor

Outline

- Motivation
- Overview of the Project
- How we split it into 3 parts
- Some “guide code” in Matlab (which is very close to pseudocode)

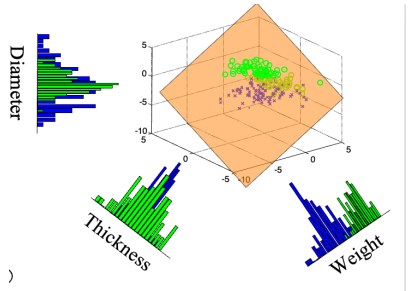
Motivation

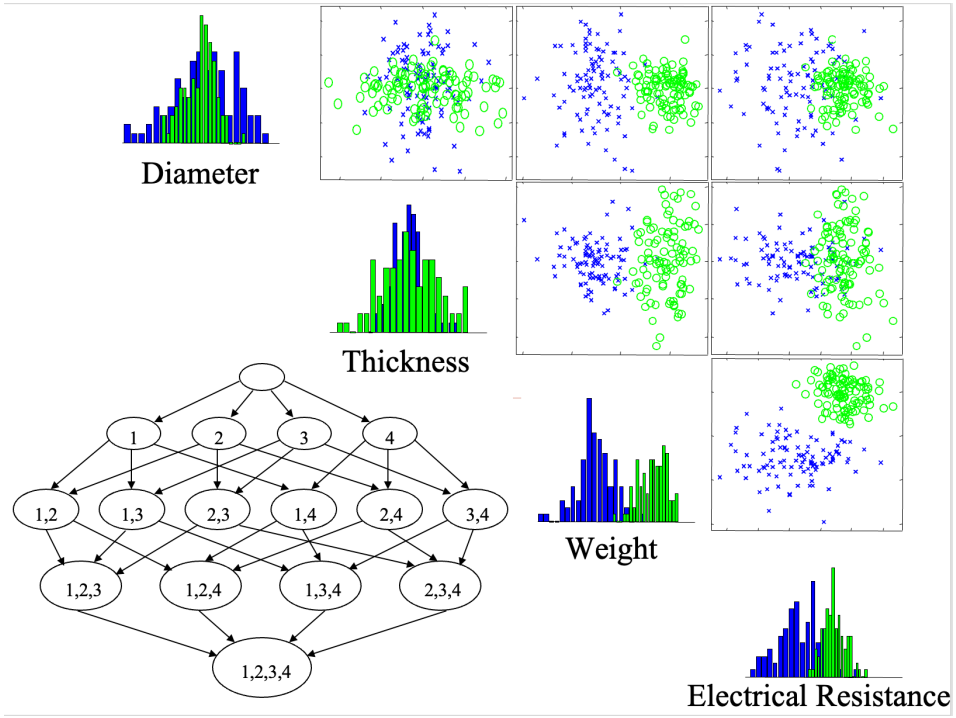
Let us revisit the original problem of classifying Canadian vs. American Quarters

Which of our features (if any) are useful?

1. Diameter
2. Thickness
3. Weight
4. Electrical Resistance

I measured these features for 50 Canadian and 50 American quarters….





You may ask:

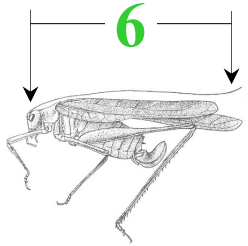
Why not just use all the features? (relevant and irrelevant)

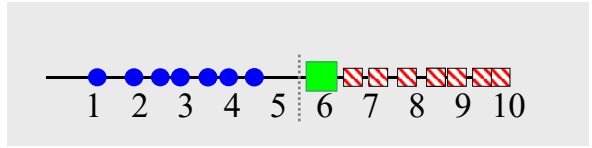
It **depends on the classifier** (linear, nearest neighbor, Bayesian, decision tree, etc)

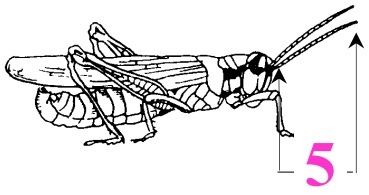
Remember that **Nearest Neighbor** Classifier was **sensitive** to **irrelevant** features
(i.e., it would be **misled** by irrelevant features)

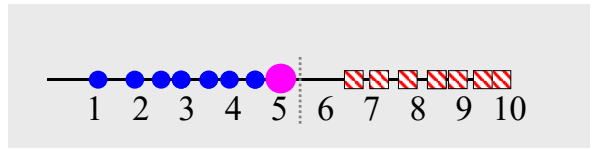
The nearest neighbor algorithm is sensitive to **irrelevant features**…

Suppose the following is true, if an insects antenna is longer than 5.5 it is a **Katydid**, otherwise it is a **Grasshopper**.

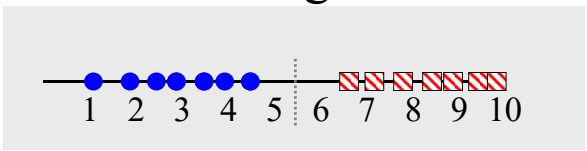




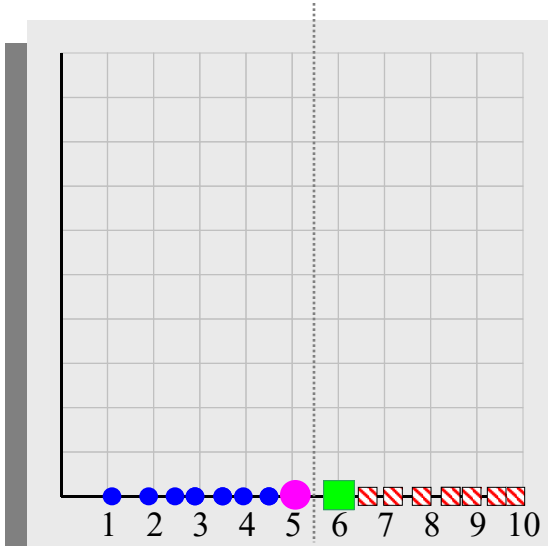


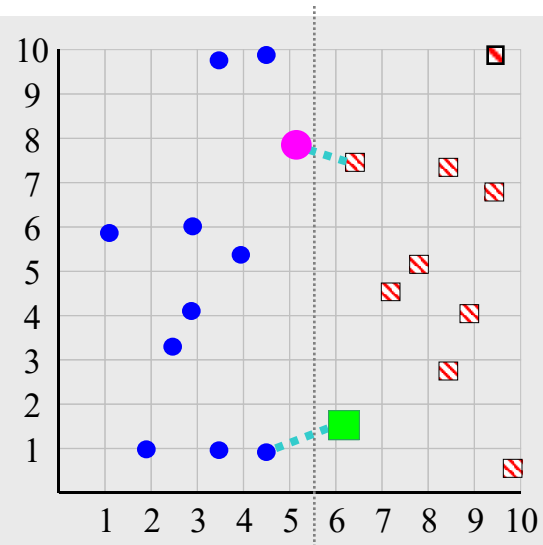


Training data



Using just the antenna length we get perfect classification!

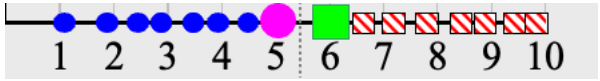




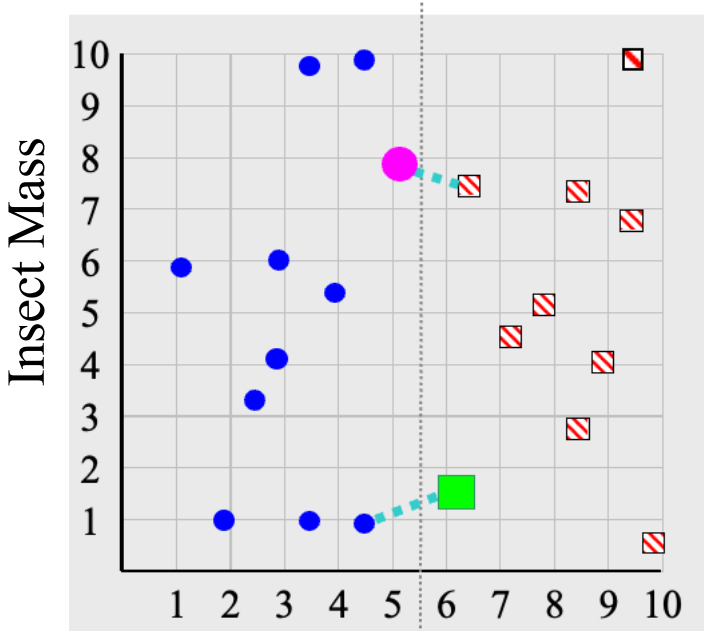
Suppose however, we add in an **irrelevant** feature, for example the insects mass.

Using both the antenna length and the insects mass with the 1-NN algorithm we get the wrong classification!

=> we need to **find out which features are relevant**
and only keep those relevant features.



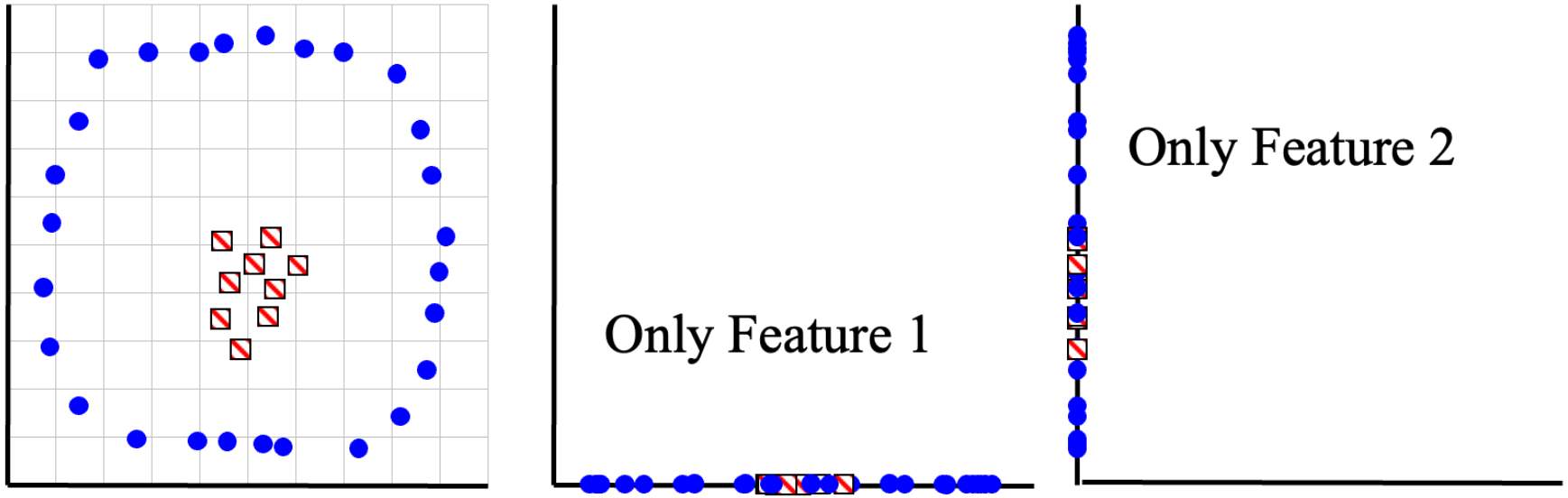
Antenna Length



Antenna Length

Antenna Length: Relevant
Insect Mass: Irrelevant and Misleads Nearest Neighbor Classifier

Let’s see another example where exactly two features are relevant.



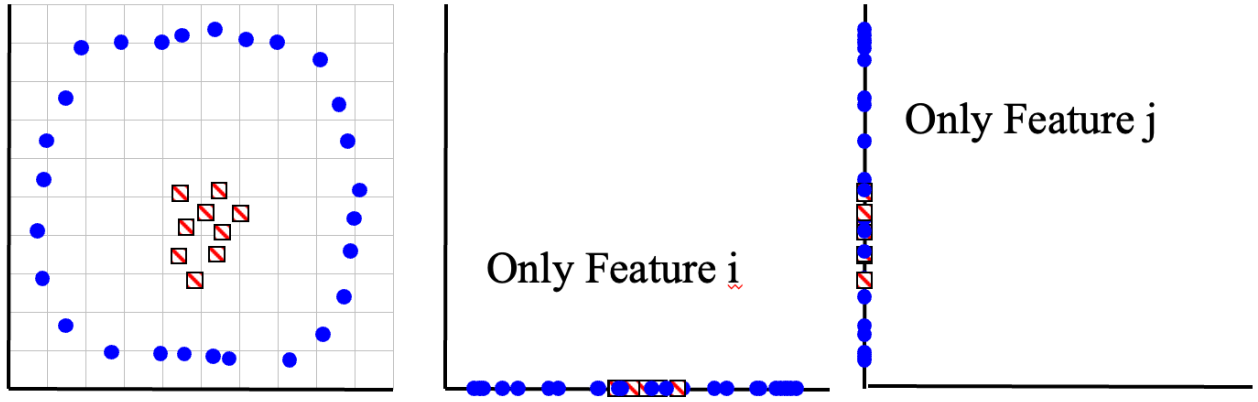
Question:

When we start with an ML problem and list all possible features, do we know which features are relevant?

Why searching over feature subsets is hard

If we have come up **with 100 features**, how do we know which feature subset to use for the classifier?

Let's say it happens (but we don't know) that **Features i and j (the X and Y below) give perfect classification**,
but all 98 of the other features are irrelevant…

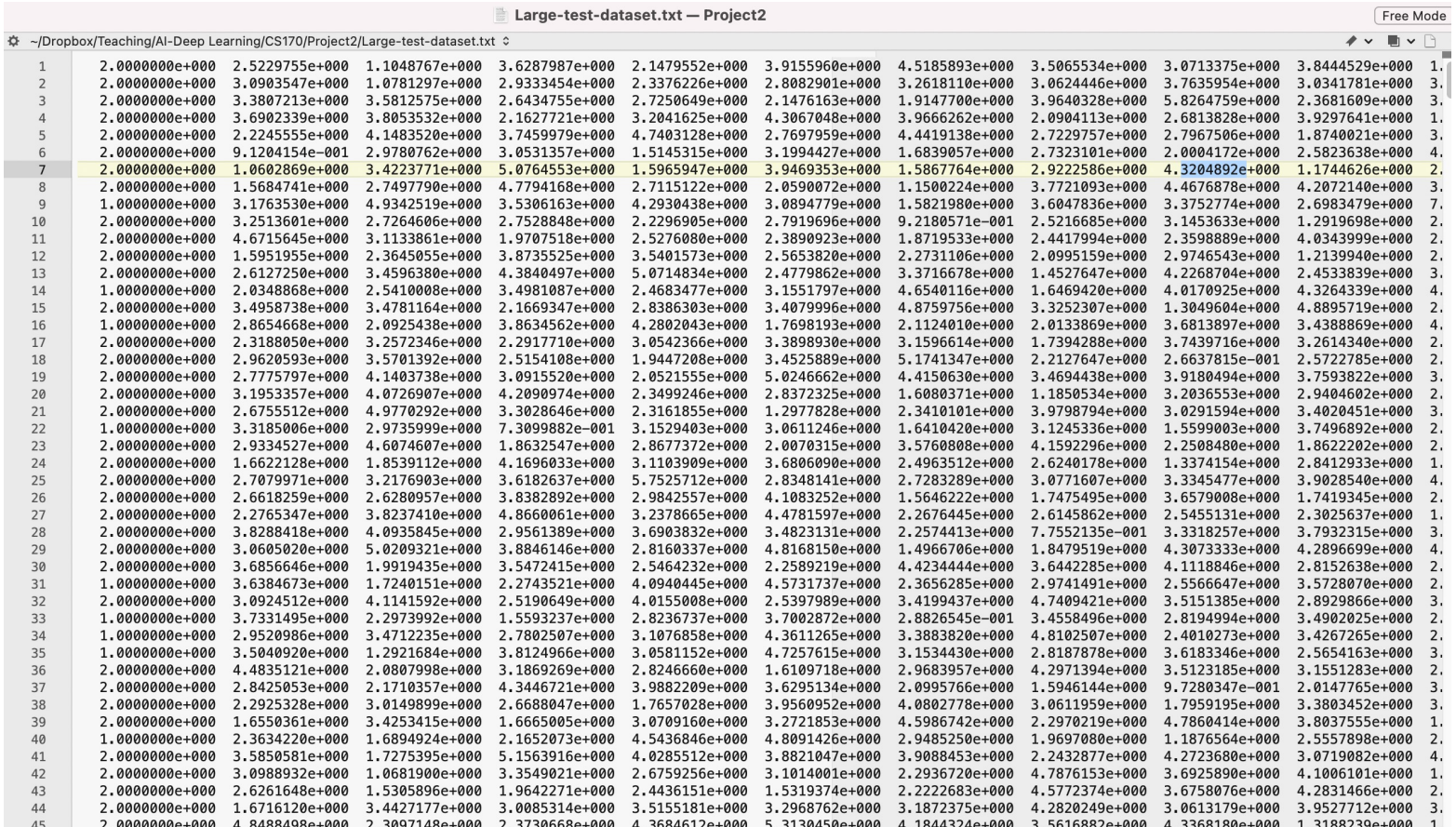


Using all 100 features will give poor results,
but so will using only Feature i alone, and so will using Feature j alone!
Of the 2^{100} –1 possible subsets of the features, **only one** really works.

Let’s see a real dataset. The one you will work with in Project2!

First column is the class (1 or 2)
All the other columns are features (40 features)
(Not all columns are shown here)

BUT We don't know which subset of these 40 features are relevant...

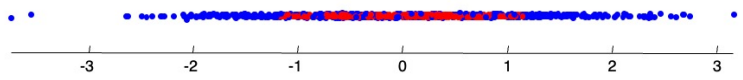


1	2.0000000e+000	2.5229755e+000	1.1048767e+000	3.6287987e+000	2.1479552e+000	3.9155960e+000	4.5185893e+000	3.5065534e+000	3.0713375e+000	3.8444529e+000
2	2.0000000e+000	3.0903547e+000	1.0781297e+000	2.9333454e+000	2.3376226e+000	2.8082901e+000	3.2618110e+000	3.0624446e+000	3.7635954e+000	3.0341781e+000
3	2.0000000e+000	3.3807213e+000	3.5812575e+000	2.6434755e+000	2.7250649e+000	2.1476163e+000	1.9147700e+000	3.9640328e+000	5.8264759e+000	2.3681609e+000
4	2.0000000e+000	3.6902339e+000	3.8053532e+000	2.1627721e+000	3.2041625e+000	4.3067048e+000	3.9666262e+000	2.0904113e+000	2.6813828e+000	3.9297641e+000
5	2.0000000e+000	2.2245555e+000	4.1483520e+000	3.7459979e+000	4.7403128e+000	2.7697959e+000	4.4419138e+000	2.7229757e+000	2.7967506e+000	1.8740021e+000
6	2.0000000e+000	9.1204154e−001	2.9780762e+000	3.0531357e+000	1.5145315e+000	3.1994427e+000	1.6839057e+000	2.7323101e+000	2.0004172e+000	2.5823638e+000
7	2.0000000e+000	1.0602869e+000	3.4223771e+000	5.0764553e+000	1.5965947e+000	3.9469353e+000	1.5867764e+000	2.9222586e+000	4.3204892e+000	1.1744626e+000
8	2.0000000e+000	1.5684741e+000	2.7497790e+000	4.7794168e+000	2.7115122e+000	2.0590072e+000	1.1500224e+000	3.7721093e+000	4.4676878e+000	4.2072140e+000
9	1.0000000e+000	3.1763530e+000	4.9342519e+000	3.5306163e+000	4.2930438e+000	3.0894779e+000	1.5821980e+000	3.6047836e+000	3.3752774e+000	2.6983479e+000
10	2.0000000e+000	3.2513601e+000	2.7264606e+000	2.7528848e+000	2.2296905e+000	2.7919696e+000	9.2180571e−001	2.5216685e+000	3.1453633e+000	1.2919698e+000
11	2.0000000e+000	4.6715645e+000	3.1133861e+000	1.9707518e+000	2.5276080e+000	2.3890923e+000	1.8719533e+000	2.4417994e+000	2.3598889e+000	4.0343999e+000
12	2.0000000e+000	1.5951955e+000	2.3645055e+000	3.8735525e+000	3.5401573e+000	2.5653820e+000	2.2731106e+000	2.0995159e+000	2.9746543e+000	1.2139940e+000
13	2.0000000e+000	2.6127250e+000	3.4596380e+000	4.3840497e+000	5.0714834e+000	2.4779862e+000	3.3716678e+000	1.4527647e+000	4.2268704e+000	2.4533839e+000
14	1.0000000e+000	2.0348868e+000	2.5410008e+000	3.4981087e+000	2.4683477e+000	3.1551797e+000	4.6540116e+000	1.6469420e+000	4.0170925e+000	4.3264339e+000
15	2.0000000e+000	3.4958738e+000	3.4781164e+000	2.1669347e+000	2.8386303e+000	3.4079996e+000	4.8759756e+000	3.3252307e+000	1.3049604e+000	4.8895719e+000
16	1.0000000e+000	2.8654668e+000	2.0925438e+000	3.8634562e+000	4.2802043e+000	1.7698193e+000	2.1124010e+000	2.0133869e+000	3.6813897e+000	3.4388869e+000
17	2.0000000e+000	2.3188050e+000	3.2572346e+000	2.2917710e+000	3.0542366e+000	3.3898930e+000	3.1596614e+000	1.7394288e+000	3.7439716e+000	3.2614340e+000
18	2.0000000e+000	2.9620593e+000	3.5701392e+000	2.5154108e+000	1.9447208e+000	3.4525889e+000	5.1741347e+000	2.2127647e+000	2.6637815e−001	2.5722785e+000
19	2.0000000e+000	2.7775797e+000	4.1403738e+000	3.0915520e+000	2.0521555e+000	5.0246662e+000	4.4150630e+000	3.4694438e+000	3.9180494e+000	3.7593822e+000
20	2.0000000e+000	3.1953357e+000	4.0726907e+000	4.2090974e+000	2.3499246e+000	2.8372325e+000	1.6080371e+000	1.1850534e+000	3.2036553e+000	2.9404602e+000
21	2.0000000e+000	2.6755512e+000	4.9770292e+000	3.3028646e+000	2.3161855e+000	1.2977828e+000	2.3410101e+000	3.9798794e+000	3.0291594e+000	3.4020451e+000
22	1.0000000e+000	3.3185006e+000	2.9735999e+000	7.3099882e−001	3.1529403e+000	3.0611246e+000	1.6410420e+000	3.1245336e+000	1.5599003e+000	3.7496892e+000
23	2.0000000e+000	2.9334527e+000	4.6074607e+000	1.8632547e+000	2.8677372e+000	2.0070315e+000	3.5760808e+000	4.1592296e+000	2.2508480e+000	1.8622202e+000
24	2.0000000e+000	1.6622128e+000	1.8539112e+000	4.1696033e+000	3.1103909e+000	3.6806090e+000	2.4963512e+000	2.6240178e+000	1.3374154e+000	2.8412933e+000
25	2.0000000e+000	2.7079971e+000	3.2176903e+000	3.6182637e+000	5.7525712e+000	2.8348141e+000	2.7283289e+000	3.0771607e+000	3.3345477e+000	3.9028540e+000
26	2.0000000e+000	2.6618259e+000	2.6280957e+000	3.8382892e+000	2.9842557e+000	4.1083252e+000	1.5646222e+000	1.7475495e+000	3.6579008e+000	1.7419345e+000
27	2.0000000e+000	2.2765347e+000	3.8237410e+000	4.8660061e+000	3.2378665e+000	4.4781597e+000	2.2676445e+000	2.6145862e+000	2.5455131e+000	2.3025637e+000
28	2.0000000e+000	3.8288418e+000	4.0935845e+000	2.9561389e+000	3.6903832e+000	3.4823131e+000	2.2574413e+000	7.7552135e−001	3.3318257e+000	3.7932315e+000
29	2.0000000e+000	3.0605020e+000	5.0209321e+000	3.8846146e+000	2.8160337e+000	4.8168150e+000	1.4966706e+000	1.8479519e+000	4.3073333e+000	4.2896699e+000
30	2.0000000e+000	3.6856646e+000	1.9919435e+000	3.5472415e+000	2.5464232e+000	2.2589219e+000	4.4234444e+000	3.6442285e+000	4.1118846e+000	2.8152638e+000
31	1.0000000e+000	3.6384673e+000	1.7240151e+000	2.2743521e+000	4.0940445e+000	4.5731737e+000	2.3656285e+000	2.9741491e+000	2.5566647e+000	3.5728070e+000
32	2.0000000e+000	3.0924512e+000	4.1141592e+000	2.5190649e+000	4.0155008e+000	2.5397989e+000	3.4199437e+000	4.7409421e+000	3.5151385e+000	2.8929866e+000
33	1.0000000e+000	3.7331495e+000	2.2973992e+000	1.5593237e+000	2.8236737e+000	3.7002872e+000	2.8826545e−001	3.4558496e+000	2.8194994e+000	3.4902025e+000
34	1.0000000e+000	2.9520986e+000	3.4712235e+000	2.7802507e+000	3.1076858e+000	4.3611265e+000	3.3883820e+000	4.8102507e+000	2.4010273e+000	3.4267265e+000
35	1.0000000e+000	3.5040920e+000	1.2921684e+000	3.8124966e+000	3.0581152e+000	4.7257615e+000	3.1534430e+000	2.8187878e+000	3.6183346e+000	2.5654163e+000
36	2.0000000e+000	4.4835121e+000	2.0807998e+000	3.1869269e+000	2.8246660e+000	1.6109718e+000	2.9683957e+000	4.2971394e+000	3.5123185e+000	3.1551283e+000
37	2.0000000e+000	2.8425053e+000	2.1710357e+000	4.3446721e+000	3.9882209e+000	3.6295134e+000	2.0995766e+000	1.5946144e+000	9.7280347e−001	2.0147765e+000
38	2.0000000e+000	2.2925328e+000	3.0149899e+000	2.6688047e+000	1.7657028e+000	3.9560952e+000	4.0802778e+000	3.0611959e+000	1.7959195e+000	3.3803452e+000
39	2.0000000e+000	1.6550361e+000	3.4253415e+000	1.6665005e+000	3.0709160e+000	3.2721853e+000	4.5986742e+000	2.2970219e+000	4.7860414e+000	3.8037555e+000
40	1.0000000e+000	2.3634220e+000	1.6894924e+000	2.1652073e+000	4.5436846e+000	4.8091426e+000	2.9485250e+000	1.9697080e+000	1.1876564e+000	2.5557898e+000
41	2.0000000e+000	3.5850581e+000	1.7275395e+000	5.1563916e+000	4.0285512e+000	3.8821047e+000	3.9088453e+000	2.2432877e+000	4.2723680e+000	3.0719082e+000
42	2.0000000e+000	3.0988932e+000	1.0681900e+000	3.3549021e+000	2.6759256e+000	3.1014001e+000	2.2936720e+000	4.7876153e+000	3.6925890e+000	4.1006101e+000
43	2.0000000e+000	2.6261648e+000	1.5305896e+000	1.9642271e+000	2.4436151e+000	1.5319374e+000	2.2222683e+000	4.5772374e+000	3.6758076e+000	4.2831466e+000
44	2.0000000e+000	1.6716120e+000	3.4427177e+000	3.0085314e+000	3.5155181e+000	3.2968762e+000	3.1872375e+000	4.2820249e+000	3.0613179e+000	3.9527712e+000
45	2.0000000e+000	4.8488498e+000	2.3097148e+000	2.3730668e+000	4.3684612e+000	5.3130450e+000	4.1844324e+000	3.5616882e+000	4.3368180e+000	1.3188239e+000

We can try to plot all feature subsets.

All single features

F1



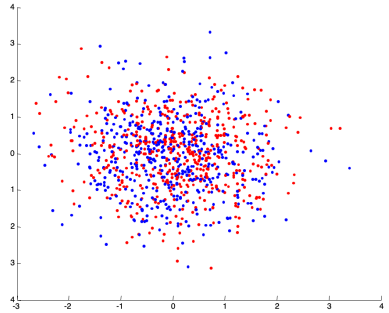
F2 …

F3 …

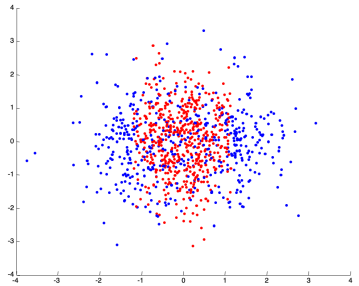
… …

All pairs of features

F1,F2



F1,F3



…

Large-test-dataset.txt — Project2

Free Mode

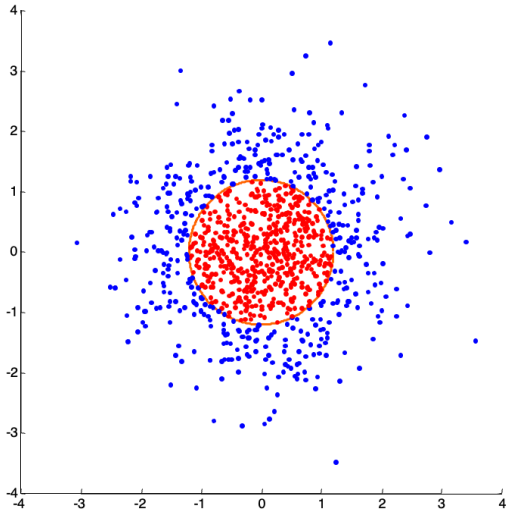
~/Dropbox/Teaching/AI-Deep Learning/CS170/Project2/Large-test-dataset.txt

1	2.0000000e+000	2.5229755e+000	1.1048767e+000	3.6287987e+000	2.1479552e+000	3.9155960e+000	4.5185893e+000	3.5065534e+000	3.0713375e+000	3.8444529e+000	1.
2	2.0000000e+000	3.0903547e+000	1.0781297e+000	2.9333454e+000	2.3376226e+000	2.8082901e+000	3.2618110e+000	3.0624446e+000	3.7635954e+000	3.0341781e+000	3.
3	2.0000000e+000	3.3807213e+000	3.5812575e+000	2.6434755e+000	2.7250649e+000	2.1476163e+000	1.9147700e+000	3.9640328e+000	5.8264759e+000	2.3681609e+000	3.
4	2.0000000e+000	3.6902339e+000	3.8053532e+000	2.1627721e+000	3.2041625e+000	4.3867048e+000	3.9666262e+000	2.0904113e+000	2.6813828e+000	3.9297641e+000	1.
5	2.0000000e+000	2.2245555e+000	4.1483520e+000	3.7459979e+000	4.7403128e+000	2.7697959e+000	4.4419138e+000	2.7229757e+000	2.7967506e+000	1.8740021e+000	3.
6	2.0000000e+000	9.1204154e−001	2.9780762e+000	3.0531357e+000	1.5145315e+000	3.1994427e+000	1.6839057e+000	2.7323101e+000	2.0004172e+000	2.5823638e+000	4.
7	2.0000000e+000	1.0602869e+000	3.4223771e+000	5.0764553e+000	1.5965947e+000	3.9469353e+000	1.5867764e+000	2.9222586e+000	4.3204892e+000	1.1744626e+000	2.
8	2.0000000e+000	1.5684741e+000	2.7497790e+000	4.7794168e+000	2.7115122e+000	2.0590072e+000	1.1500224e+000	3.7721093e+000	4.4676878e+000	4.2072140e+000	3.
9	1.0000000e+000	3.1763530e+000	4.9342519e+000	3.5306163e+000	4.2930438e+000	3.0894779e+000	1.5821980e+000	3.6047836e+000	3.3752774e+000	2.6983479e+000	7.
10	2.0000000e+000	3.2513601e+000	2.7264606e+000	2.7528848e+000	2.2296905e+000	2.7919696e+000	9.2180571e−001	2.5216685e+000	3.1453633e+000	1.2919698e+000	2.
11	2.0000000e+000	4.6715645e+000	3.1133861e+000	1.9707518e+000	2.5276080e+000	2.3890923e+000	1.8719533e+000	2.4417994e+000	2.3598889e+000	4.0343999e+000	2.
12	2.0000000e+000	1.5951955e+000	2.3645055e+000	3.8735525e+000	3.5401573e+000	2.5653820e+000	2.2731106e+000	2.0995159e+000	2.9746543e+000	1.2139940e+000	2.
13	2.0000000e+000	2.6127250e+000	3.4596380e+000	4.3040497e+000	5.0714834e+000	2.4779062e+000	3.3716678e+000	1.4527647e+000	4.2260704e+000	2.4533839e+000	3.
14	1.0000000e+000	2.0348868e+000	2.5410008e+000	3.4981087e+000	2.4683477e+000	3.1551797e+000	4.6540116e+000	1.6469420e+000	4.0170925e+000	4.3264339e+000	4.
15	2.0000000e+000	3.4958738e+000	3.4781164e+000	2.1669347e+000	2.8386303e+000	3.4079996e+000	4.8759756e+000	3.3252307e+000	1.3049604e+000	4.8895719e+000	2.
16	1.0000000e+000	2.8654668e+000	2.0925438e+000	3.8634562e+000	4.2802043e+000	1.7698193e+000	2.1124010e+000	2.0133869e+000	3.6813897e+000	3.4388869e+000	4.
17	2.0000000e+000	2.3188050e+000	3.2572346e+000	2.2917710e+000	3.0542366e+000	3.3898930e+000	3.1596614e+000	1.7394288e+000	3.7439716e+000	3.2614340e+000	2.
18	2.0000000e+000	2.9620593e+000	3.5701392e+000	2.5154108e+000	1.9447208e+000	3.4525889e+000	5.1741347e+000	2.2127647e+000	2.6637815e−001	2.5722785e+000	2.
19	2.0000000e+000	2.7775797e+000	4.1403738e+000	3.0915520e+000	2.0521555e+000	5.0246662e+000	4.4150630e+000	3.4694438e+000	3.9180494e+000	3.7593822e+000	3.
20	2.0000000e+000	3.1953357e+000	4.0726907e+000	4.2090974e+000	2.3499246e+000	2.8372325e+000	1.6080371e+000	1.1850534e+000	3.2036553e+000	2.9404602e+000	2.
21	2.0000000e+000	2.6755512e+000	4.9770292e+000	3.3028646e+000	2.3161855e+000	1.2977828e+000	2.3410101e+000	3.9798794e+000	3.0291594e+000	3.4020451e+000	3.
22	1.0000000e+000	3.3185006e+000	2.9735999e+000	7.3099882e−001	3.1529403e+000	3.0611246e+000	1.6410420e+000	3.1245336e+000	1.5599003e+000	3.7496892e+000	2.
23	2.0000000e+000	2.9334527e+000	4.6074607e+000	1.8632547e+000	2.8677372e+000	2.0070315e+000	3.5760808e+000	4.1592296e+000	2.2508480e+000	1.8622202e+000	2.
24	2.0000000e+000	1.6622128e+000	1.8539112e+000	4.1696033e+000	3.1103909e+000	3.6806090e+000	2.4963512e+000	2.6240178e+000	1.3374154e+000	2.8412933e+000	1.
25	2.0000000e+000	2.7079971e+000	3.2176903e+000	3.6182637e+000	5.7525712e+000	2.8348141e+000	2.7283289e+000	3.0771607e+000	3.3345477e+000	3.9028540e+000	4.
26	2.0000000e+000	2.6618259e+000	2.6280957e+000	3.8382892e+000	2.9842557e+000	4.1083252e+000	1.5646222e+000	1.7475495e+000	3.6579008e+000	1.7419345e+000	2.
27	2.0000000e+000	2.2765347e+000	3.8237410e+000	4.8660061e+000	3.2378665e+000	4.4781597e+000	2.2676445e+000	2.6145862e+000	2.5455131e+000	2.3025637e+000	1.
28	2.0000000e+000	3.8288418e+000	4.0935845e+000	2.9561389e+000	3.6903832e+000	3.4823131e+000	2.2574413e+000	7.7552135e−001	3.3318257e+000	3.7932315e+000	3.
29	2.0000000e+000	3.0605020e+000	5.0209321e+000	3.8846146e+000	2.8160337e+000	4.8168150e+000	1.4966706e+000	1.8479519e+000	4.3073333e+000	4.2896699e+000	4.
30	2.0000000e+000	3.6856646e+000	1.9919435e+000	3.5472415e+000	2.5464232e+000	2.2589219e+000	4.4234444e+000	3.6442285e+000	4.1118846e+000	2.8152638e+000	2.
31	2.0000000e+000	3.6384673e+000	1.7240151e+000	2.2743521e+000	4.0940445e+000	4.5731737e+000	2.3656285e+000	2.9741491e+000	2.5566647e+000	3.5728070e+000	2.
32	2.0000000e+000	3.8924512e+000	4.1141592e+000	2.5190649e+000	4.0155008e+000	2.5397989e+000	3.4199437e+000	4.7409421e+000	3.5151385e+000	2.8929866e+000	3.
33	1.0000000e+000	3.7331495e+000	2.2973992e+000	1.5593237e+000	2.8236737e+000	3.7002872e+000	2.8826545e−001	3.4558496e+000	2.8194994e+000	3.4902025e+000	2.
34	1.0000000e+000	2.9520986e+000	3.4712235e+000	2.7802507e+000	3.1076858e+000	4.3611265e+000	3.3883820e+000	4.8102507e+000	2.4010273e+000	3.4267265e+000	2.
35	1.0000000e+000	3.5040920e+000	1.2921684e+000	3.8124966e+000	3.0581152e+000	4.7257615e+000	3.1534430e+000	2.8187878e+000	3.6183346e+000	2.5654163e+000	3.
36	2.0000000e+000	4.4835121e+000	2.0807998e+000	3.1869269e+000	2.8246660e+000	1.6109718e+000	2.9683957e+000	4.2971394e+000	3.5123185e+000	3.1551283e+000	2.
37	2.0000000e+000	2.8425053e+000	2.1710357e+000	4.3446721e+000	3.9882209e+000	3.6295134e+000	2.0995766e+000	1.5946144e+000	9.7280347e−001	2.0147765e+000	3.
38	2.0000000e+000	2.2925328e+000	3.0149899e+000	2.6688047e+000	1.7657028e+000	3.9560952e+000	4.0802778e+000	3.0611959e+000	1.7959195e+000	3.3803452e+000	3.
39	2.0000000e+000	1.6550361e+000	3.4253415e+000	1.6665005e+000	3.0709160e+000	3.2721853e+000	4.5986742e+000	2.2970219e+000	4.7860414e+000	3.8037555e+000	1.
40	1.0000000e+000	2.3634220e+000	1.6894924e+000	2.1652073e+000	4.5436846e+000	4.8091426e+000	2.9485250e+000	1.9697080e+000	1.1876564e+000	2.5557898e+000	2.
41	2.0000000e+000	3.5850581e+000	1.7275395e+000	5.1563916e+000	4.0285512e+000	3.8821047e+000	3.9008453e+000	2.2432877e+000	4.2723680e+000	3.0719082e+000	4.
42	2.0000000e+000	3.0988932e+000	1.0681900e+000	3.3549021e+000	2.6759256e+000	3.1014001e+000	2.2936720e+000	4.7876153e+000	3.6925890e+000	4.1006101e+000	1.
43	2.0000000e+000	2.6261648e+000	1.5305896e+000	1.9642271e+000	2.4436151e+000	1.5319374e+000	2.2222683e+000	4.5772374e+000	3.6758076e+000	4.2831466e+000	2.
44	2.0000000e+000	1.6716120e+000	3.4427177e+000	3.0085314e+000	3.5155181e+000	3.2968762e+000	3.1872375e+000	4.2820249e+000	3.0613179e+000	3.9527712e+000	3.
45	2.0000000e+000	4.8488498e+000	2.3097148e+000	2.3730668e+000	4.3684612e+000	5.3130450e+000	4.1844324e+000	3.5616882e+000	4.3368180e+000	1.3188239e+000	1.
46	2.0000000e+000	9.6848001e−001	2.8536694e+000	3.3099160e+000	3.9077819e+000	2.2326963e+000	1.9521184e+000	4.0186234e+000	1.6848337e+000	3.2216915e+000	2.
47	2.0000000e+000	4.8468742e+000	1.9201624e+000	2.6766054e+000	3.7259019e+000	6.1041787e+000	2.6952097e+000	4.7071912e+000	3.2778315e+000	4.3225825e+000	2.

And all other subsets with 3 features, all subsets with 4 features, … all subsets with 40 features

Why searching over feature subsets is hard

We can? try all the subsets and eventually find the best subset (with 2 features) that give us the best accuracy:

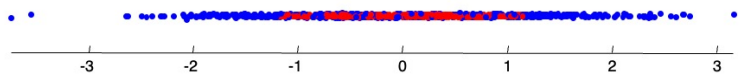


Using all 40 features will give poor results,
but so will using only Feature i alone, and so will using Feature j alone!
Of the 2^{40} –1 possible subsets of the features, **only one** really works.

We can try to plot all feature subsets.

All single features

F1



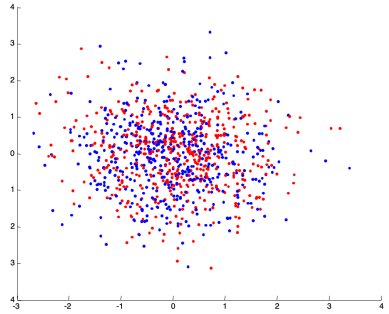
F2 …

F3 …

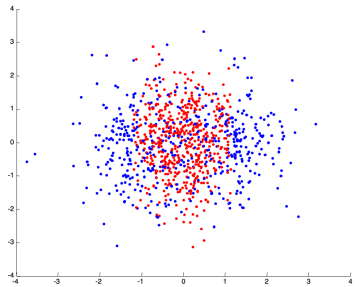
… …

All pairs of features

F1,F2



F1,F3



…

Large-test-dataset.txt — Project2

Free Mode

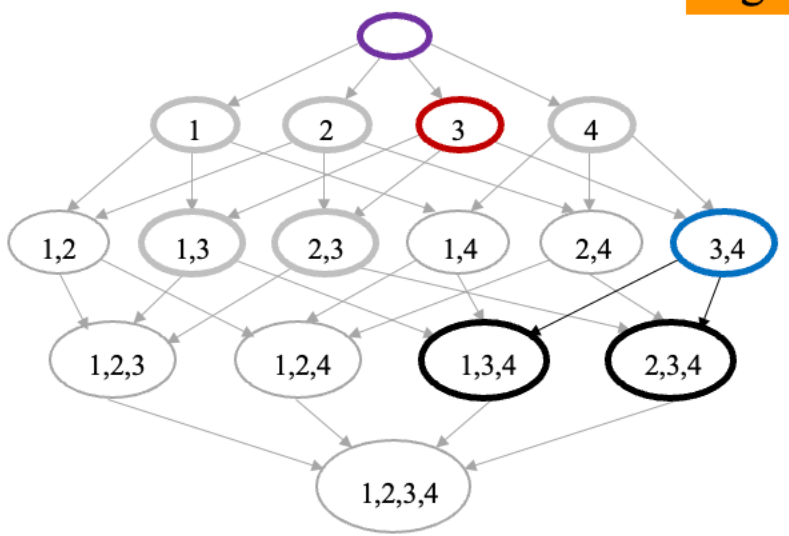
~/Dropbox/Teaching/AI-Deep Learning/CS170/Project2/Large-test-dataset.txt

1	2.0000000e+000	2.5229755e+000	1.1048767e+000	3.6287987e+000	2.1479552e+000	3.9155960e+000	4.5185893e+000	3.5065534e+000	3.0713375e+000	3.8444529e+000	1.
2	2.0000000e+000	3.0903547e+000	1.0781297e+000	2.9333454e+000	2.3376226e+000	2.8082901e+000	3.2618110e+000	3.0624446e+000	3.7635954e+000	3.0341781e+000	3.
3	2.0000000e+000	3.3807213e+000	3.5812575e+000	2.6434755e+000	2.7250649e+000	2.1476163e+000	1.9147700e+000	3.9640328e+000	5.8264759e+000	2.3681609e+000	3.
4	2.0000000e+000	3.6902339e+000	3.8053532e+000	2.1627721e+000	3.2041625e+000	4.3867048e+000	3.9666262e+000	2.0904113e+000	2.6813828e+000	3.9297641e+000	1.
5	2.0000000e+000	2.2245555e+000	4.1483520e+000	3.7459979e+000	4.7403128e+000	2.7697959e+000	4.4419138e+000	2.7229757e+000	2.7967506e+000	1.8740021e+000	3.
6	2.0000000e+000	9.1204154e−001	2.9780762e+000	3.0531357e+000	1.5145315e+000	3.1994427e+000	1.6839057e+000	2.7323101e+000	2.0004172e+000	2.5823638e+000	4.
7	2.0000000e+000	1.0602869e+000	3.4223771e+000	5.0764553e+000	1.5965947e+000	3.9469353e+000	1.5867764e+000	2.9222586e+000	4.3204892e+000	1.1744626e+000	2.
8	2.0000000e+000	1.5684741e+000	2.7497790e+000	4.7794168e+000	2.7115122e+000	2.0590072e+000	1.1500224e+000	3.7721093e+000	4.4676878e+000	4.2072140e+000	3.
9	1.0000000e+000	3.1763530e+000	4.9342519e+000	3.5306163e+000	4.2930438e+000	3.0894779e+000	1.5821980e+000	3.6047836e+000	3.3752774e+000	2.6983479e+000	7.
10	2.0000000e+000	3.2513601e+000	2.7264606e+000	2.7528848e+000	2.2296905e+000	2.7919696e+000	9.2180571e−001	2.5216685e+000	3.1453633e+000	1.2919698e+000	2.
11	2.0000000e+000	4.6715645e+000	3.1133861e+000	1.9707518e+000	2.5276080e+000	2.3890923e+000	1.8719533e+000	2.4417994e+000	2.3598889e+000	4.0343999e+000	2.
12	2.0000000e+000	1.5951955e+000	2.3645055e+000	3.8735525e+000	3.5401573e+000	2.5653820e+000	2.2731106e+000	2.0995159e+000	2.9746543e+000	1.2139940e+000	2.
13	2.0000000e+000	2.6127250e+000	3.4596380e+000	4.3040497e+000	5.0714834e+000	2.4779862e+000	3.3716678e+000	1.4527647e+000	4.2260704e+000	2.4533839e+000	3.
14	1.0000000e+000	2.0348868e+000	2.5410008e+000	3.4981087e+000	2.4683477e+000	3.1551797e+000	4.6540116e+000	1.6469420e+000	4.0170925e+000	4.3264339e+000	4.
15	2.0000000e+000	3.4958738e+000	3.4781164e+000	2.1669347e+000	2.8386303e+000	3.4079996e+000	4.8759756e+000	3.3252307e+000	1.3049604e+000	4.8895719e+000	2.
16	1.0000000e+000	2.8654668e+000	2.0925438e+000	3.8634562e+000	4.2802043e+000	1.7698193e+000	2.1124010e+000	2.0133869e+000	3.6813897e+000	3.4388869e+000	4.
17	2.0000000e+000	2.3188050e+000	3.2572346e+000	2.2917710e+000	3.0542366e+000	3.3898930e+000	3.1596614e+000	1.7394288e+000	3.7439716e+000	3.2614340e+000	2.
18	2.0000000e+000	2.9620593e+000	3.5701392e+000	2.5154108e+000	1.9447208e+000	3.4525889e+000	5.1741347e+000	2.2127647e+000	2.6637815e−001	2.5722785e+000	2.
19	2.0000000e+000	2.7775797e+000	4.1403738e+000	3.0915520e+000	2.0521555e+000	5.0246662e+000	4.4150630e+000	3.4694438e+000	3.9180494e+000	3.7593822e+000	3.
20	2.0000000e+000	3.1953357e+000	4.0726907e+000	4.2090974e+000	2.3499246e+000	2.8372325e+000	1.6080371e+000	1.1850534e+000	3.2036553e+000	2.9404602e+000	2.
21	2.0000000e+000	2.6755512e+000	4.9770292e+000	3.3028646e+000	2.3161855e+000	1.2977828e+000	2.3410101e+000	3.9798794e+000	3.0291594e+000	3.4020451e+000	3.
22	1.0000000e+000	3.3185006e+000	2.9735999e+000	7.3099882e−001	3.1529403e+000	3.0611246e+000	1.6410420e+000	3.1245336e+000	1.5599003e+000	3.7496892e+000	2.
23	2.0000000e+000	2.9334527e+000	4.6074607e+000	1.8632547e+000	2.8677372e+000	2.0070315e+000	3.5760808e+000	4.1592296e+000	2.2508480e+000	1.8622202e+000	2.
24	2.0000000e+000	1.6622128e+000	1.8539112e+000	4.1696033e+000	3.1103909e+000	3.6806090e+000	2.4963512e+000	2.6240178e+000	1.3374154e+000	2.8412933e+000	1.
25	2.0000000e+000	2.7079971e+000	3.2176903e+000	3.6182637e+000	5.7525712e+000	2.8348141e+000	2.7283289e+000	3.0771607e+000	3.3345477e+000	3.9028540e+000	4.
26	2.0000000e+000	2.6618259e+000	2.6280957e+000	3.8382892e+000	2.9842557e+000	4.1083252e+000	1.5646222e+000	1.7475495e+000	3.6579008e+000	1.7419345e+000	2.
27	2.0000000e+000	2.2765347e+000	3.8237410e+000	4.8660061e+000	3.2378665e+000	4.4781597e+000	2.2676445e+000	2.6145862e+000	2.5455131e+000	2.3025637e+000	1.
28	2.0000000e+000	3.8288418e+000	4.0935845e+000	2.9561389e+000	3.6903832e+000	3.4823131e+000	2.2574413e+000	7.7552135e−001	3.3318257e+000	3.7932315e+000	3.
29	2.0000000e+000	3.0605020e+000	5.0209321e+000	3.8846146e+000	2.8160337e+000	4.8168150e+000	1.4966706e+000	1.8479519e+000	4.3073333e+000	4.2896699e+000	4.
30	2.0000000e+000	3.6856646e+000	1.9919435e+000	3.5472415e+000	2.5464232e+000	2.2589219e+000	4.4234444e+000	3.6442285e+000	4.1118846e+000	2.8152638e+000	2.
31	2.0000000e+000	3.6384673e+000	1.7240151e+000	2.2743521e+000	4.0940445e+000	4.5731737e+000	2.3656285e+000	2.9741491e+000	2.5566647e+000	3.5728070e+000	2.
32	2.0000000e+000	3.8924512e+000	4.1141592e+000	2.5190649e+000	4.0155008e+000	2.5397989e+000	3.4199437e+000	4.7409421e+000	3.5151385e+000	2.8929866e+000	3.
33	1.0000000e+000	3.7331495e+000	2.2973992e+000	1.5593237e+000	2.8236737e+000	3.7002872e+000	2.8826545e−001	3.4558496e+000	2.8194994e+000	3.4902025e+000	2.
34	1.0000000e+000	2.9520986e+000	3.4712235e+000	2.7802507e+000	3.1076858e+000	4.3611265e+000	3.3883820e+000	4.8102507e+000	2.4010273e+000	3.4267265e+000	2.
35	1.0000000e+000	3.5040920e+000	1.2921684e+000	3.8124966e+000	3.0581152e+000	4.7257615e+000	3.1534430e+000	2.8187878e+000	3.6183346e+000	2.5654163e+000	3.
36	2.0000000e+000	4.4835121e+000	2.0807998e+000	3.1869269e+000	2.8246660e+000	1.6109718e+000	2.9683957e+000	4.2971394e+000	3.5123185e+000	3.1551283e+000	2.
37	2.0000000e+000	2.8425053e+000	2.1710357e+000	4.3446721e+000	3.9882209e+000	3.6295134e+000	2.0995766e+000	1.5946144e+000	9.7280347e−001	2.0147765e+000	3.
38	2.0000000e+000	2.2925328e+000	3.0149899e+000	2.6688047e+000	1.7657028e+000	3.9560952e+000	4.0802778e+000	3.0611959e+000	1.7959195e+000	3.3803452e+000	3.
39	2.0000000e+000	1.6550361e+000	3.4253415e+000	1.6665005e+000	3.0709160e+000	3.2721853e+000	4.5986742e+000	2.2970219e+000	4.7860414e+000	3.8037555e+000	1.
40	1.0000000e+000	2.3634220e+000	1.6894924e+000	2.1652073e+000	4.5436846e+000	4.8091426e+000	2.9485250e+000	1.9697080e+000	1.1876564e+000	2.5557898e+000	2.
41	2.0000000e+000	3.5850581e+000	1.7275395e+000	5.1563916e+000	4.0285512e+000	3.8821047e+000	3.9008453e+000	2.2432877e+000	4.2723680e+000	3.0719082e+000	4.
42	2.0000000e+000	3.0988932e+000	1.0681900e+000	3.3549021e+000	2.6759256e+000	3.1014001e+000	2.2936720e+000	4.7876153e+000	3.6925890e+000	4.1006101e+000	1.
43	2.0000000e+000	2.6261648e+000	1.5305896e+000	1.9642271e+000	2.4436151e+000	1.5319374e+000	2.2222683e+000	4.5772374e+000	3.6758076e+000	4.2831466e+000	2.
44	2.0000000e+000	1.6716120e+000	3.4427177e+000	3.0085314e+000	3.5155181e+000	3.2968762e+000	3.1872375e+000	4.2820249e+000	3.0613179e+000	3.9527712e+000	3.
45	2.0000000e+000	4.8488498e+000	2.3097148e+000	2.3730668e+000	4.3684612e+000	5.3130450e+000	4.1844324e+000	3.5616882e+000	4.3368180e+000	1.3188239e+000	1.
46	2.0000000e+000	9.6848001e−001	2.8536694e+000	3.3099160e+000	3.9077819e+000	2.2326963e+000	1.9521184e+000	4.0186234e+000	1.6848337e+000	3.2216915e+000	2.
47	2.0000000e+000	4.8468742e+000	1.9201624e+000	2.6766054e+000	3.7259019e+000	6.1041787e+000	2.6952097e+000	4.7071912e+000	3.2778315e+000	4.3225825e+000	2.

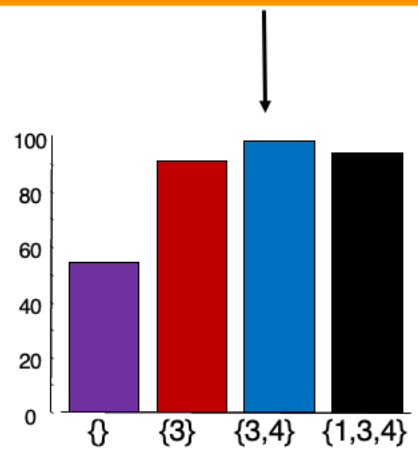
BUT CAN WE PLOT ALL 2^{40} –1 POSSIBLE SUBSETS?

Greedy Forward Section

Initial state: Empty Set: No features
Operators: Add a feature.
Evaluation Function: **Leave-one-out**



Highest-scoring node (feature subset)

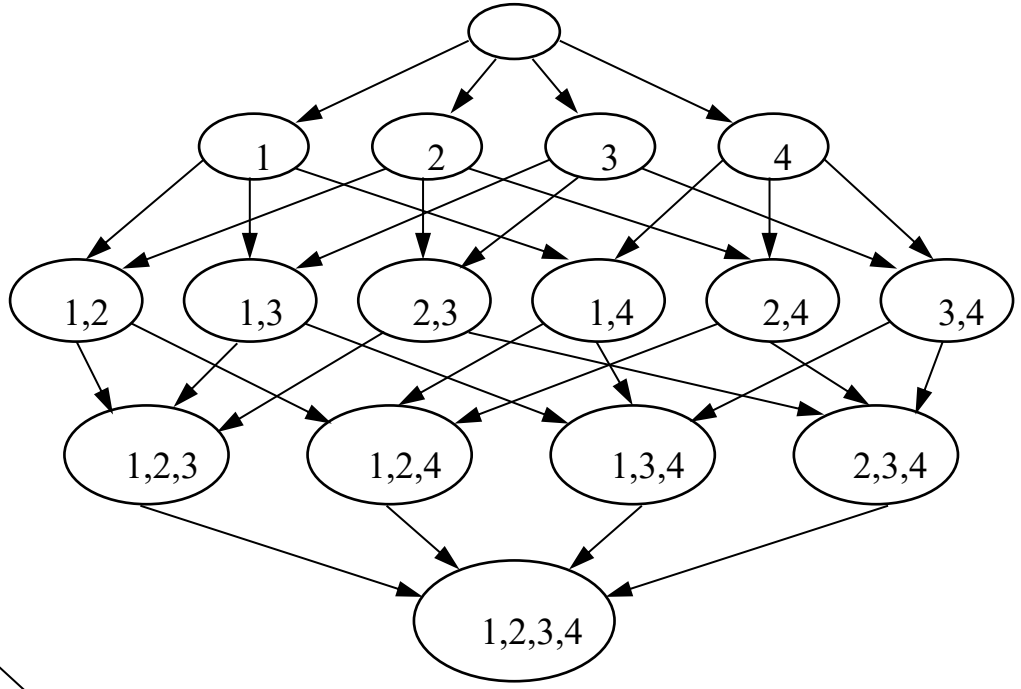


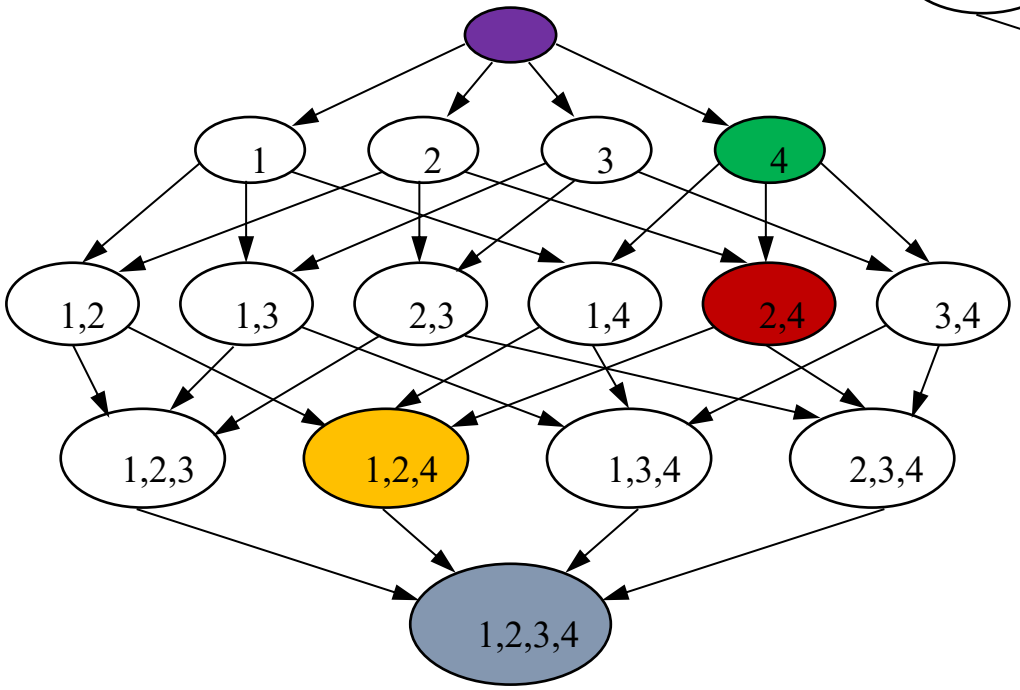
Outline

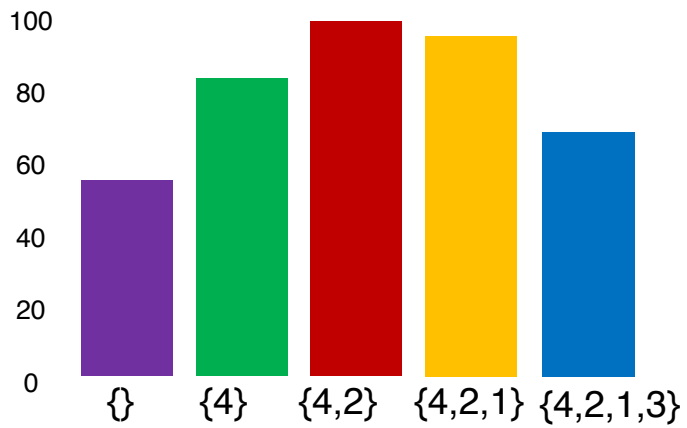
- Motivation
- Overview of the Project
- How we split it into 3 parts
- Some “guide code” in Matlab (which is very close to pseudocode)

Problem Formulation:

Given a dataset with N features, we want to **find the subset of features** that results in the **highest accuracy** (when an NN classifier is used).



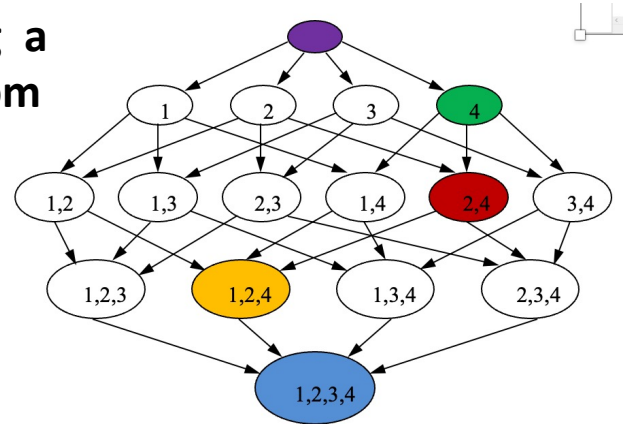


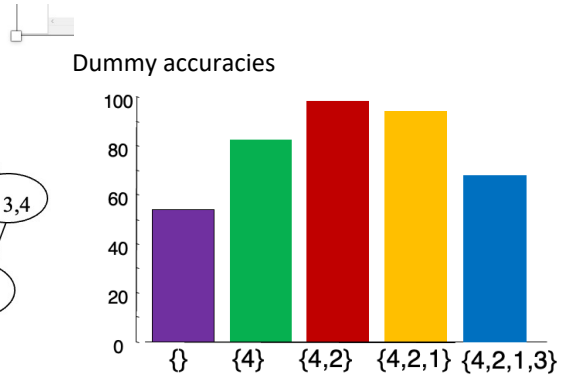


Outline

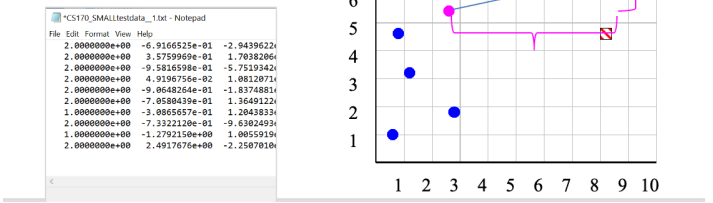
- Motivation
- Overview of the Project
- How we split it into 3 parts
- Some “guide code” in Matlab (which is very close to pseudocode)

Part I: Implement search only (using a dummy validator that returns random numbers as accuracies!)

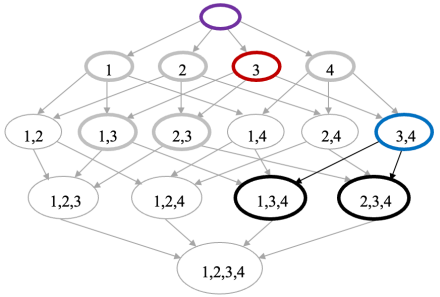


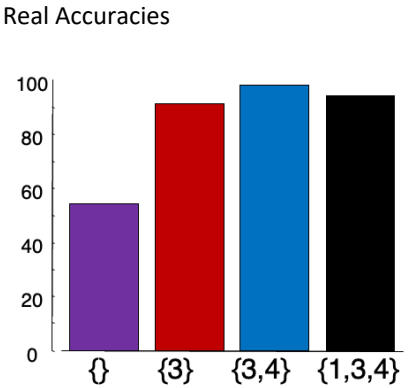


Part II: Implement NN classifier (too easy) and the leave-one-out validator that use the dataset (easy) => Now we can calculate the actual accuracy of the NN classifier on a dataset (with a particular subset of features).



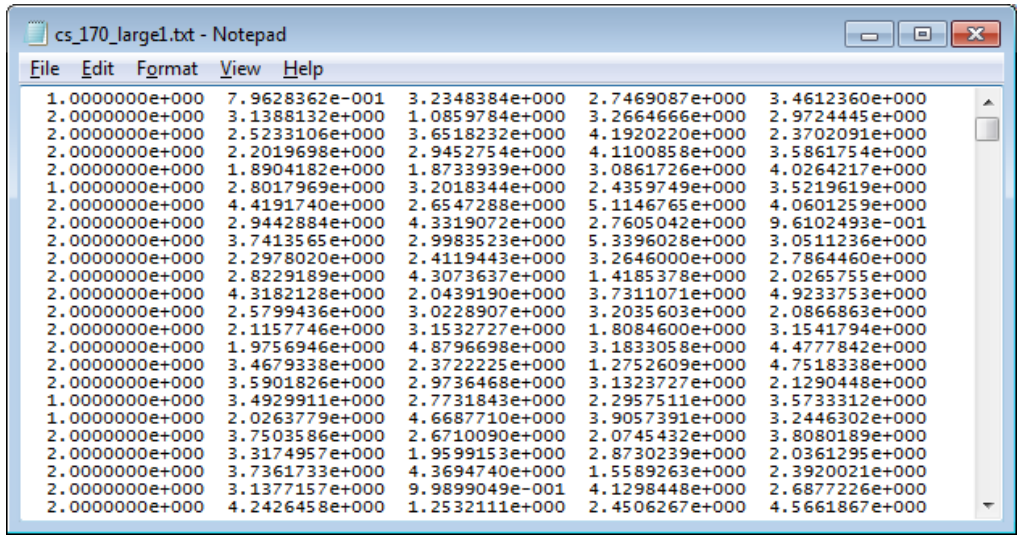
Part III: Replace the dummy Use the leave-one-out validator in the search algorithm from part I





This is a toy figure. You are going to have (#columns-1) features.

Input: The dataset which is a text file like this:



Each row is a data point and each column corresponds to a different feature

- Number of Rows: Number of data points (instances)
- Number of columns: Number of features

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

Outline

- Motivation
- Overview of the Project
- How we split it into 3 parts
- Some “guide code” in Matlab (which is very close to pseudocode)

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

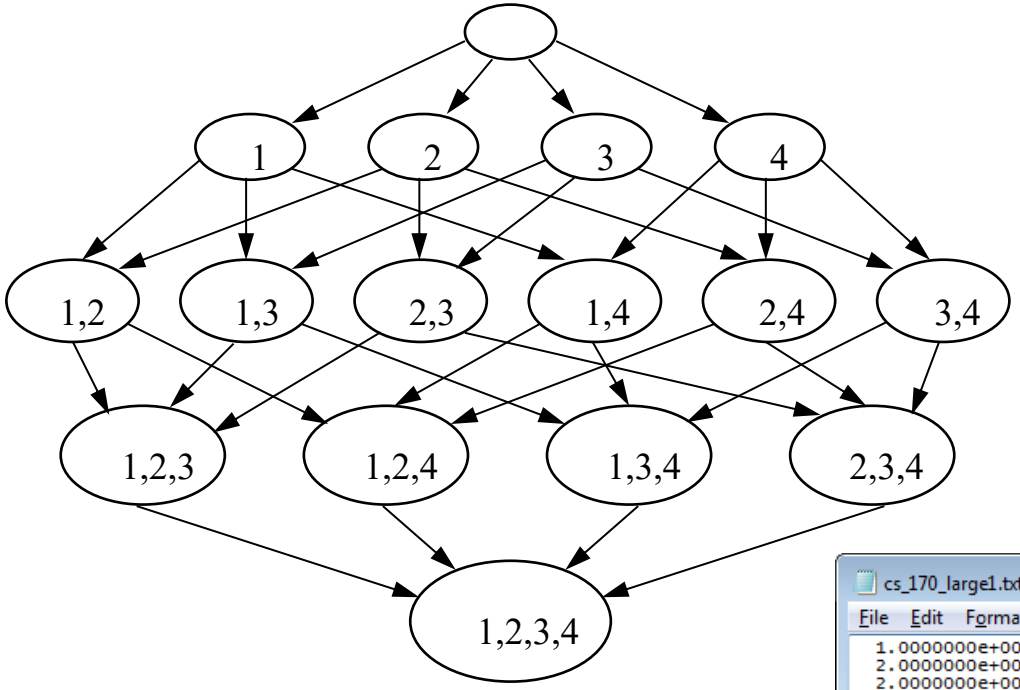
To do this, I wrote a stub function that just returns a random number

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

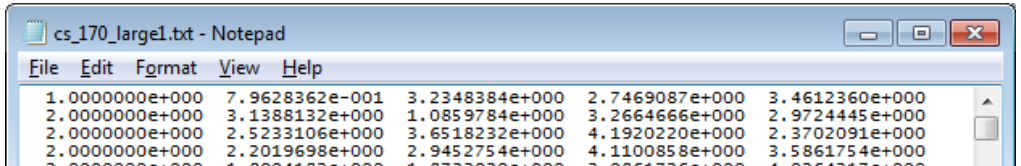
I will use this in my search algorithm, and only when I am 100% sure that search works, will I “fill in” the full leave-one-out-cross-validation code.

```
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…

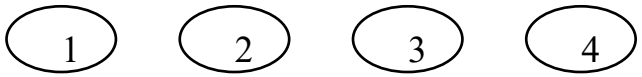



```
EDU>> feature_search_demo(mydata)
On the 1th level of the search tree
On the 2th level of the search tree
On the 3th level of the search tree
On the 4th level of the search tree
```



```
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).

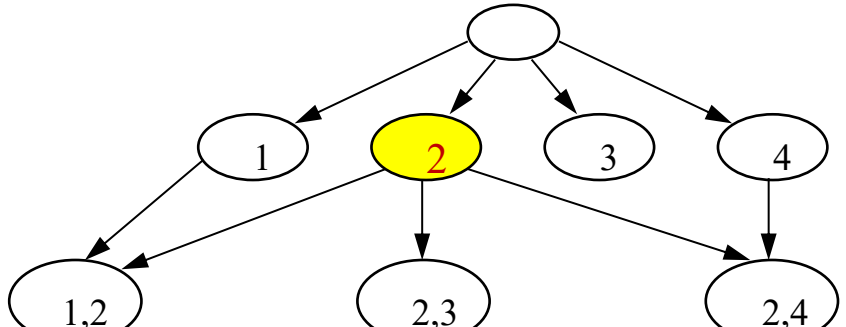
```
function  feature_search_demo(data)

      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);

              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
```

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);

              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)



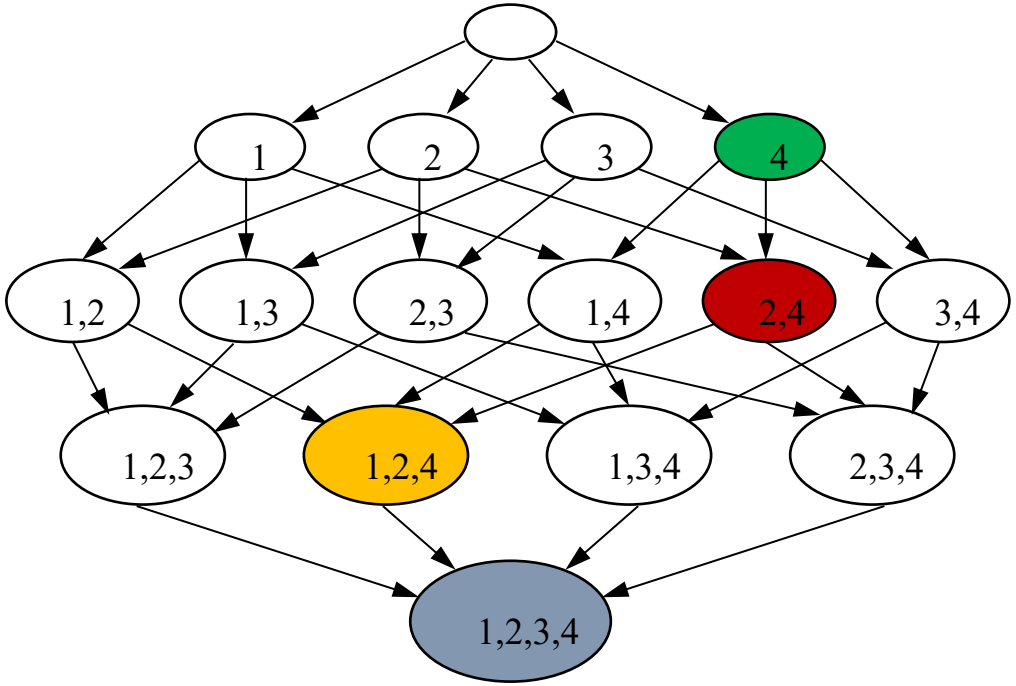
…We need an IF statement in the inner loop that says “only consider adding this feature, if it was not already added”

```
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.
Later (in Part III) 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
```

Note that the previous code was for guidance.
You will need to print the accuracies at each level and submit your code that outputs a trace like this:

Welcome to Bertie Woosters (change this to your name) Feature Selection Algorithm.

Please enter total number of features: **4**

Type the number of the algorithm you want to run.

- Forward Selection
- Backward Elimination
- Bertie's Special Algorithm.

1

Using no features and "random" evaluation, I get an accuracy of 55.4%

Beginning search.

Using feature(s) {1} accuracy is 35.4%
Using feature(s) {2} accuracy is 56.7%
Using feature(s) {3} accuracy is 41.4%
Using feature(s) {4} accuracy is 28.5%

Feature set {2} was best, accuracy is 56.7%

Using feature(s) {1,2} accuracy is 58.9%
Using feature(s) {3,2} accuracy is 40.4%
Using feature(s) {4,2} accuracy is 58.1%

Feature set {1,2} was best, accuracy is 58.9%

Using feature(s) {3,1,2} accuracy is 60.1%
Using feature(s) {4,1,2} accuracy is 76.4%

Feature set {4,1,2} was best, accuracy is 76.4%

Using feature(s) {1,2,4,3} accuracy is 73.1%

(Warning, Accuracy has decreased!)
Finished search!! The best feature subset is {4,1,2}, which has an accuracy of 76.4%