Tutorial 10:
ID3 Decision Trees and Naïve Bayes Classification

Question 1

a) Building the decision tree
Start by choosing top attribute.

Gain(A) = I(p/(p+n),n/(p+n)) - Remainder(A).

Can ignore the first term (common to all attributes) and just find the attribute with the smallest remainder term.

Remainder(A) = sum (i over all possible values of attribute A) (p_i+n_i)/(p+n) I((p_i/(p_i+n_i), (n_i/(p_i+n_i))

Identities worth remembering:
I(0,1) = 0 = I(1,0)
I(0.5,0.5) = 1
I(x,1-x)=I(1-x,x) .

p = 6 (# of cases with PlayTennis=Yes), n = 4 (# of cases with PlayTennis=No)

Outlook:
p_Sunny = 1 ; n_Sunny = 3
p_Overcast = 2 ; n_Overcast = 0
p_Rain = 3 ; n_Rain = 1

Remainder(Outlook) = (4/10) I(1/4,3/4) + (2/10) I(2/2,0/2) + (4/10) I(3/4,1/4)
= 0.4*(-(1/4)*log2(1/4)-(3/4)*log2(3/4)) + 0 + (4/10)*(-(3/4)*log2(3/4)-(1/4)*log2(1/4))
= 0.649

Temperature:
p_Hot = 1 ; n_Hot = 2
p_Mild = 2 ; n_Mild = 1
p_Cool = 3 ; n_Cool = 1

Remainder(Temperature) = (3/10) I(1/3,2/3) + (3/10) I(2/3,1/3) + (4/10) I(3/4,1/4)
= 0.6*(-(1/3)*log2(1/3)-(2/3)*log2(2/3)) + (4/10)*(-(3/4)*log2(3/4)-(1/4)*log2(1/4))
= 0.875

Humidity:
p_High = 2 ; n_High = 3
p_Normal = 4 ; n_Normal = 1

Remainder(Humidity) = (5/10) I(2/5,3/5) + (5/10) I(4/5,1/5)
= 0.5*(-(2/5)*log2(2/5)-(3/5)*log2(3/5)) + (5/10)*(-(4/5)*log2(4/5)-(1/5)*log2(1/5))
= 0.846

Wind:
p_Weak = 5 ; n_Weak = 2

$p_{Strong} = 1$; $n_{Strong} = 2$

Remainder(Wind) = (7/10) I(5/7,2/7) + (3/10) I(1/3,2/3)
= 0.7*(-(5/7)*log2(5/7)-(2/7)*log2(2/7)) + 0.3*(-(1/3)*log2(1/3)-(2/3)*log2(2/3))
= 0.880

Outlook has the smallest remainder and so is the best attribute to place at the top of the ID3 decision tree.

Building remainder of tree:

Due to the split on Outlook, the dataset is now split into 3 parts, each of which may need one or more splits to perfectly classify the training examples into PlayTennis=Yes or No.

The split data is shown below.

Day Outlook Temperature Humidity Wind PlayTennis

D1 Sunny Hot High Weak No
D2 Sunny Hot High Strong No
D8 Sunny Mild High Weak No
D9 Sunny Cool Normal Weak Yes
-> needs another split to perfectly classify this.
Without doing any calculations, it is pretty clear that Humidity is the best one to use.
Why? Well in all 4 cases, High Humidity -> No and Normal Humidity -> Yes.

Admittedly Temperature works just as well, but it offers 3 options and we only have 4 training patterns here, so I am going to say that choosing an attribute with a smaller number of splits is better when two or more attributes are equally good. In ID3, technically, there is no difference and one could choose arbitrarily between Humidity and Temperature. In ID3's successors: C4.5 and C5, I believe they penalise using more splits, so these would choose Humidity.

Wind doesn't work very well. Strong Wind gets 1 No, Weak Wind gets 2 No's and 1 Yes.

An interesting issue: Imagine Humidity had 3 possible values (3rd being: Low), but only two appear in this subset of the data. What if we split on Humidity, but got a test case with e.g. Outlook=Sunny and Humidity=Low ? According to Figure 18.5 (Decision Tree Learning algorithm pseudo code), we should include a Humidity=Low branch in the tree and give it the value Default (which we can choose), ie: Yes or No. Instead of making an arbitrary choice, I suggest using the majority class of the examples just above that branch, but to a large extent, we'd be guessing.

Other subsets after split on Outlook:

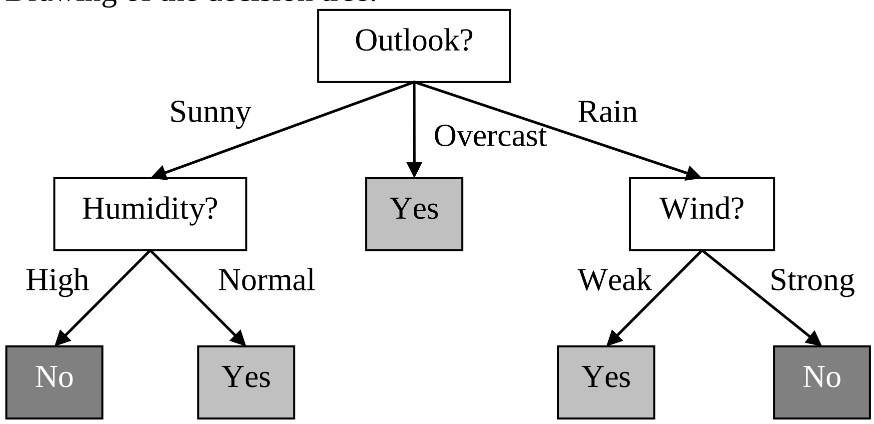
D3 Overcast Hot High Weak Yes
D7 Overcast Cool Normal Strong Yes
-> no further branching needed here: just classify all cases reaching this leaf as Yes.

D4 Rain Mild High Weak Yes
D5 Rain Cool Normal Weak Yes
D6 Rain Cool Normal Strong No
D10 Rain Mild Normal Weak Yes

We've also been lucky here. The attribute Wind lines up perfectly with the response (PlayTennis). The other attributes are definitely not as good.
So we branch here on Wind, letting Weak Wind -> Yes and Strong Wind -> No.

The resulting tree will classify all the training patterns correctly.

Drawing of the decision tree:



b) There is no noise in the data, so there is no need to average over the outcomes. You can simply label each leaf node with yes or no as above.

c) Classifying the test data:

D11 -> Yes
D12 -> Yes
D13 -> Yes
D14 -> No

-> the decision tree got them all right.

d)
So there are no failing test cases and no evidence of needing to adjust anything.

Note that if you chose temperature instead of humidity, D11 will be misclassified.

Question 2

For this tutorial, we can just construct the model and its prediction as needed in response to each test pattern. I.e. there is no need to build the full Naive Bayes model for every possible test pattern. However, if we had a lot of data and wanted to use the trained system extensively, it would be worth estimating each of the possible P(attribute=value|class) combinations before using the model on test cases.

The true value of each test case is given in brackets. Remember: we estimate all of the required probabilities from the training data unless given better evidence of their value from somewhere else. We do not use any of the test data to estimate the probabilities.

P(PlayTennis = Yes) = 6/10 = 0.6 .
P(PlayTennis = No) = 1-P(PlayTennis = Yes) = 0.4

D11 Sunny Mild Normal Strong (Yes)

Naive Bayes model:
P(PlayTennis=Yes|D11 (without PlayTennis value !))
= P(PlayTennis=Yes|Outlook=Sunny,Temperature=Mild,Humidity=Normal,Wind=Strong)
= P(Outlook=Sunny,Temperature=Mild,Humidity=Normal,Wind=Strong|PlayTennis=Yes)
P(PlayTennis = Yes)/C

C, above is a constant = P(Outlook=Sunny,Temperature=Mild,Humidity=Normal,Wind=Strong) . We could estimate this from the data using Naive Bayes assumptions (decomposing it into a product of 4 terms for each condition), but there is no need - it will disappear in the renormalisation of the probabilities so that they sum to 1, so we can ignore the constant.

Continuing the equation above using the Naive Bayes model:
= P(Outlook=Sunny,Temperature=Mild,Humidity=Normal,Wind=Strong|PlayTennis=Yes)
P(PlayTennis = Yes)/C
= P(Outlook=Sunny|PlayTennis=Yes) P(Temperature=Mild|PlayTennis=Yes)
P(Humidity=Normal|PlayTennis=Yes)
P(Wind=Strong|PlayTennis=Yes) P(PlayTennis = Yes)/C

All of the above should now be estimated from the training data.
= (1/6) * (2/6) * (4/6) * (1/6) * 0.6/C = 0.00370/C

The alternative classification would be PlayTennis=No .
We now find the probability of this under the Naive Bayes model.

P(PlayTennis=No|Outlook=Sunny,Temperature=Mild,Humidity=Normal,Wind=Strong)
= P(Outlook=Sunny|PlayTennis=No) P(Temperature=Mild|PlayTennis=No)
P(Humidity=Normal|PlayTennis=No)
P(Wind=Strong|PlayTennis=No) P(PlayTennis = No)/C

All of the above should now be estimated from the training data.
= (3/4) * (1/4) * (1/4) * (2/4) * 0.4/C = 0.00936/C

From the above, we can see that
P(PlayTennis=No|D11) is larger than P(PlayTennis=Yes|D11) , and so the prediction should be No.
That happens to disagree with the supplied label, but you expect to see some errors on test data.

Renormalising the values so that they sum to 1:

P(PlayTennis=Yes|D11) = 0.00370/(0.00936+0.00370) = 0.283
So P(PlayTennis=No|D11) = 1-0.283 = 0.717 .

D12 Overcast Mild High Strong (Yes)

P(PlayTennis=Yes|D12)
= P(Outlook=Overcast|PlayTennis=Yes) P(Temperature=Mild|PlayTennis=Yes)
P(Humidity=High|PlayTennis=Yes)
P(Wind=Strong|PlayTennis=Yes) P(PlayTennis = Yes)/C
= (2/6) * (2/6) * (2/6) * (1/6) * 0.6/C
= 0.00370/C

Notice that we had estimated P(Temperature=Mild|PlayTennis=Yes), P(Wind=Strong|PlayTennis=Yes) and P(PlayTennis = Yes) previously and so can use these values again. We also got the same raw number (ignoring different constant) for P(PlayTennis=Yes|D11). Important: the constant C here (for D12) is different to the previous one (for D11) - you have to renormalise properly each time.

P(PlayTennis=No|D12)
= P(Outlook=Overcast|PlayTennis=No) P(Temperature=Mild|PlayTennis=No)
P(Humidity=High|PlayTennis=No)
P(Wind=Strong|PlayTennis=No) P(PlayTennis = No)/C
= (0/4) * (?/4) * (?/4) * (?/4) * 0.4/C
= 0.

No need to work out the ?'s once we see a zero.

Renormalising gives us
P(PlayTennis=Yes|D12) = 0.00370/(0.00370+0) = 1 (as you probably guessed).

So our prediction is Yes, which happens to be right.

D13 Overcast Hot Normal Weak (Yes)

P(PlayTennis=Yes|D13)
= P(Outlook=Overcast|PlayTennis=Yes) P(Temperature=Hot|PlayTennis=Yes)
P(Humidity=Normal|PlayTennis=Yes)
P(Wind=Weak|PlayTennis=Yes) P(PlayTennis = Yes)/C
= (2/6) * (1/6) * (4/6) * (5/6) * 0.6/C
= 0.0185/C

P(PlayTennis=No|D13)
= P(Outlook=Overcast|PlayTennis=No) P(Temperature=Hot|PlayTennis=No)
P(Humidity=Normal|PlayTennis=No)
P(Wind=Weak|PlayTennis=No) P(PlayTennis = No)/C
= (0/4) * (?/4) * (?/4) * (?/4) * 0.4/C
= 0

-> so renormalising gives
P(PlayTennis=Yes|D13) = 1. Our prediction is Yes, which happens to be right.

D14 Rain Mild High Strong (No)

P(PlayTennis=Yes|D14)
= P(Outlook=Rain|PlayTennis=Yes) P(Temperature=Mild|PlayTennis=Yes)
P(Humidity=High|PlayTennis=Yes)
P(Wind=Strong|PlayTennis=Yes) P(PlayTennis = Yes)/C
= (3/6) * (2/6) * (2/6) * (1/6) * 0.6/C
= 0.00556/C

P(PlayTennis=No|D14)
= P(Outlook=Rain|PlayTennis=No) P(Temperature=Mild|PlayTennis=No)
P(Humidity=High|PlayTennis=No)
P(Wind=Strong|PlayTennis=No) P(PlayTennis = No)/C

= (1/4) * (1/4) * (3/4) * (2/4) * 0.4/C
= 0.00938/C

P(PlayTennis=No|D14) is larger, so our prediction is No, which is also right.
Renormalising to find out what the model says the probability of that answer is:

P(PlayTennis=No|D14) = 0.00938/(0.00938+0.00556) = 0.628.