RAZAVET Mael 30/01/2014

Data Mining – Exercise 4

Question 1:

Degree of freedom = 1, as we have only two classes.
Significance level = 95%
=> Threshold = 3.84

A= Income		Observed frequency for class Loan		
Value of A	Interval	Y	N	Total
12	12 <= A < 13	1	1	2
13	13 <= A < 14	1	0	1
14	14 <= A <= 16	2	0	2
16	16 <= A <18	1	0	1
18	18 <= A < 19	0	1	1
19	19 <= A <21	0	1	1
21	21 <= A < 22	0	1	1
22	22 <= A < 24	0	1	1
24	24 <= A < 25	0	2	2
25	25 <= A < 32	1	1	2
32	32 <= A < 33	1	0	1
33	33 <= A < 37	2	0	2
37	37 <= A < 46	0	1	1
46	46 <= A < 53	0	1	1
53	53 <= A	0	1	1

Initial frequency table with intervals added

We calculate the chi2 of each adjacent interval.
Here is an example of how to calculate chi2 for the first two adjacent intervals.

	Yes			No			Total observed
	o	e	Val*	o	e	Val*	
12	1	1.33	0.0819	1	0.667	0.166	2
13	1	0.667	0.166	0	0.333	0.222	1
Total	2			1			3

Then, we sum up every Val* values to compute the chi2, which gives us for this example:

$$\chi^2 = 0.639$$

A= Income		Observed frequency for class Loan			
Value of A	Interval	Y	N	Total	Chi2
12	12 <= A < 13	1	1	2	0.639
13	13 <= A < 14	1	0	1	0
14	14 <= A <= 16	2	0	2	0
16	16 <= A <18	1	0	1	2
18	18 <= A < 19	0	1	1	0
19	19 <= A <21	0	1	1	0
21	21 <= A < 22	0	1	1	0
22	22 <= A < 24	0	1	1	0
24	24 <= A < 25	0	2	2	1.33
25	25 <= A < 32	1	1	2	0.639
32	32 <= A < 33	1	0	1	0
33	33 <= A < 37	2	0	2	2.56
37	37 <= A < 46	0	1	1	0
46	46 <= A < 53	0	1	1	0
53	53 <= A	0	1	1	

We can now consider the intervals having the lowest chi2 values (0 here) and compare this value with the threshold, which is 3.89. If this lower, we merge the adjacent intervals.

A= Income		Observed frequency for class Loan			
Value of A	Interval	Y	N	Total	Chi2
12	12 <= A < 13	1	1	2	1.95
13	13 <= A < 18	4	0	4	10
18	18 <= A < 25	0	6	6	2.3
25	25 <= A < 32	1	1	2	1.7
32	32 <= A < 37	3	0	3	6
37	37 <= A	0	3	3	

Next iteration: We compare the lowest chi2 to the threshold.

A= Income		Observed frequency for class Loan			
Value of A	Interval	Y	N	Total	Chi2
12	12 <= A < 18	5	1	6	7.02
18	18 <= A < 32	1	7	8	7.12
32	32 <= A < 37	3	0	3	6
37	37 <= A	0	3	3	

The last four intervals have chi2 values, which are greater than the threshold, which means the algorithm stops.
Finally, the final intervals are:
12 <= A < 18
18 <= A < 32
32 <= A < 37
37 <= A

Question 2:

```
loan = read.csv('Loan.csv')
disc=function(dataset, N){
 for(attribute in 1:length(names(dataset))){
  if(is.numeric(dataset[,attribute]))
  {
        width <- (max(dataset[,attribute]) - min(dataset[,attribute]))/N
        dataset[,attribute] <- cut(dataset[,attribute], breaks = seq(min(dataset[,attribute]),
max(dataset[,attribute]), by = width), include.lowest=TRUE)
  }
 }
 return (dataset)
}
loanDisc=disc(loan, 4)
loanDisc returns
```

	Income	Loan
1	[12,22.2]	Y
2	[12,22.2]	Y
3	[12,22.2]	Y
4	[12,22.2]	N
5	[12,22.2]	Y
6	[12,22.2]	Y
7	[12,22.2]	N
8	(32.5,42.8]	Y
9	[12,22.2]	N
10	(22.2,32.5]	N
11	(42.8,53]	N
12	(42.8,53]	N
13	(22.2,32.5]	N
14	[12,22.2]	N
15	(22.2,32.5]	N
16	(22.2,32.5]	Y
17	(32.5,42.8]	Y
18	(32.5,42.8]	N
19	[12,22.2]	N
20	(22.2,32.5]	Y

NB: To get the number of bins instead of the intervals, we can use "labels=FALSE" as parameter of the cut function.

As we can see, the three equal-width intervals are:
12 <= Income <= 22.2
22.2 < Income <= 32.5
32.5 < Income <= 42.8
42.8 < Income <= 53

Here are the tables of frequencies of the two methods:

A= Income		Observed frequency for class Loan		
Value of A	Interval	Y	N	Total
12	12 <= A <= 22.2	5	5	10
22.2	22.2 < A < 32.5	2	3	5
32.5	32.5 < A <= 42.8	2	1	3
42.8	42.8 < A <= 53	0	2	2

A= Income		Observed frequency for class Loan			
Value of A	Interval	Y	N	Total	Chi2
12	12 <= A < 18	5	1	6	7.02
18	18 <= A < 32	1	7	8	7.12
32	32 <= A < 37	3	0	3	6
37	37 <= A	0	3	3	

So, the frequencies are differently distributed, but we can say that the chi merge algorithm would distribute the data more equally than the equal-width method.

Question 3:

Level1:

$$E(Play) = -\frac{9}{14}log_2\left(\frac{9}{14}\right) - \frac{5}{14}log_2\left(\frac{5}{14}\right) = 0.94$$

		Play			
		Yes	No		Entropy
Outlook	Sunny	2	3	5	0.971
	Overcast	4	0	4	0
	Rain	2	3	5	0.971
				14	

$$E(Outlook) = \frac{5}{14}\left[-\frac{2}{5}log_2\left(\frac{2}{5}\right) - \frac{3}{5}log_2\left(\frac{3}{5}\right)\right] + \frac{4}{14}\left[-\frac{4}{4}log_2\left(\frac{4}{4}\right)\right] + \frac{5}{14}\left[-\frac{2}{5}log_2\left(\frac{2}{5}\right) - \frac{3}{5}log_2\left(\frac{3}{5}\right)\right] = 0.693$$

$$Gain(Outlook) = E(Play) - E(Outlook) = 0.94 - 0.693 = 0.247$$

		Play			
		Yes	No		Entropy
Temp	Hot	3	2	5	0.971
	Mild	3	2	5	0.971
	Cool	3	1	4	0.811
				14	

$$E(Temp) = \frac{5}{14}\left[-\frac{2}{5}log_2\left(\frac{2}{5}\right) - \frac{3}{5}log_2\left(\frac{3}{5}\right)\right] * 2 + \frac{4}{14}\left[-\frac{3}{4}log_2\left(\frac{3}{4}\right) - \frac{1}{4}log_2(\frac{1}{4})\right] = 0.925$$

$$Gain(Temp) = E(Play) - E(Temp) = 0.94 - 0.925 = 0.015$$

		Play			
		Yes	No		Entropy
Humidity	High	3	3	6	1
	Normal	6	2	8	0.811
				14	

$$E(Humidity) = \frac{6}{14}\left[-\frac{3}{6}log_2\left(\frac{3}{6}\right) * 2\right] + \frac{8}{14}\left[-\frac{6}{8}log_2\left(\frac{6}{8}\right) - \frac{2}{8}log_2(\frac{2}{8})\right] = 0.892$$

$$Gain(Humidity) = E(Play) - E(Humidity) = 0.94 - 0.892 = 0.048$$

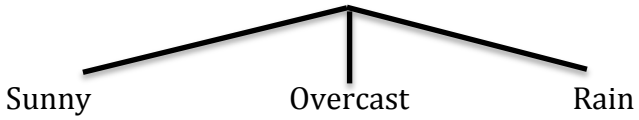
		Play			
		Yes	No		Entropy
Wind	Strong	4	3	7	0.98
	Weak	5	2	7	0.86
				14	

$$E(Wind) = \frac{7}{14}\left[-\frac{4}{7}log_2\left(\frac{4}{7}\right) - \frac{3}{7}log_2\left(\frac{3}{7}\right)\right] + \frac{7}{14}\left[-\frac{5}{7}log_2\left(\frac{5}{7}\right) - \frac{2}{7}log_2(\frac{2}{7})\right] = 0.924$$

$$Gain(Wind) = E(Play) - E(Wind) = 0.94 - 0.924 = 0.016$$

We, then, select the attribute “Outlook” to be the top node of the tree because it has the highest Information Gain.

Outlook



Level 2:

		Sunny			
		Yes	No		Entropy
Humidity	High	0	2	2	0
	Normal	2	1	3	0.918
				5	

$$E(Sunny) = 0.971$$

$$E(Humidity) = \frac{-2log_2(2) + 2log_2(2) - 2log_2(2) + 3log_2(3)}{5} = 0.55$$

$$Gain(Sunny, Humidity) = E(Sunny) - E(Humidity) = 0.971 - 0.55 = 0.42$$

We apply the same formulas and we obtain:

		Sunny			
		Yes	No		Entropy
Temp	Hot	1	2	3	0.918
	Mild	0	1	1	0
	Cool	1	0	1	0
				5	

$$E(Temp) = \frac{-2log_2(2) + 3log_2(3)}{5} = 0.55$$

$$Gain(Sunny, Temp) = 0.52$$

		Sunny			
		Yes	No		Entropy
Wind	Weak	1	2	3	0.918
	Strong	1	1	2	0

				5	

$$E(Wind) = \frac{-2log_2(2) + 2log_2(2) + 3log_2(3)}{5} = 0.951$$

$$Gain(Sunny, Wind) = 0.019$$

This means that for the attribute value Sunny, the most relevant node below is the attribute Humidity.

		Rain			
		Yes	No		Entropy
Temp	Hot	0	0	0	0
	Mild	2	1	3	0.918
	Cool	1	1	2	0
				5	

$$Gain(Rain, Temp) = 0.02$$

		Rain			
		Yes	No		Entropy
Wind	Weak	2	0	2	0
	Strong	1	2	3	0.918
				5	

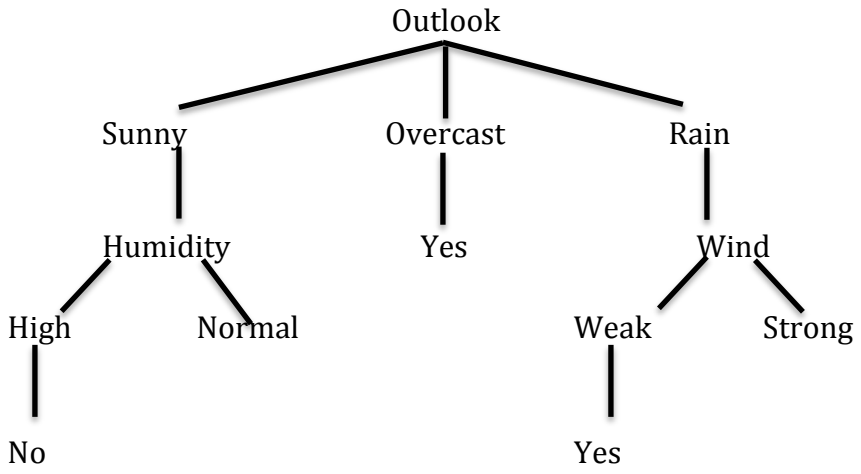
$$Gain(Rain, Wind) = 0.42$$

		Rain			
		Yes	No		Entropy
Humidity	High	0	1	15	0
	Normal	3	1	4	0.811
				5	

$$Gain(Rain, Humidity) = 0.32$$

This means that for the attribute value Rain, the most relevant node below is the attribute Wind.
We notice that for the slit on the attribute Humidity, the entropy of the value High equals to 0, which means this is a leaf and this is always ending to a “No” classification.
We can make the same observation for the value “Weak” over the attribute Wind.

However, for the attribute Overcast, as all its instances are labeled as “yes”, this means its entropy equals 0, then, this is a leaf. We cannot get more information by splitting on this attribute.
This gives us,



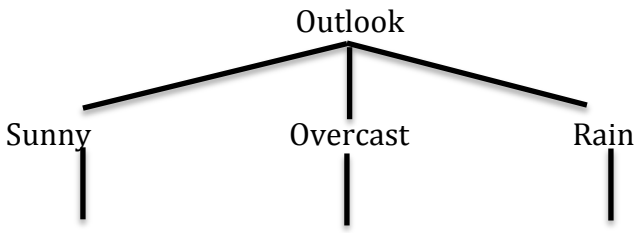
Level 3:

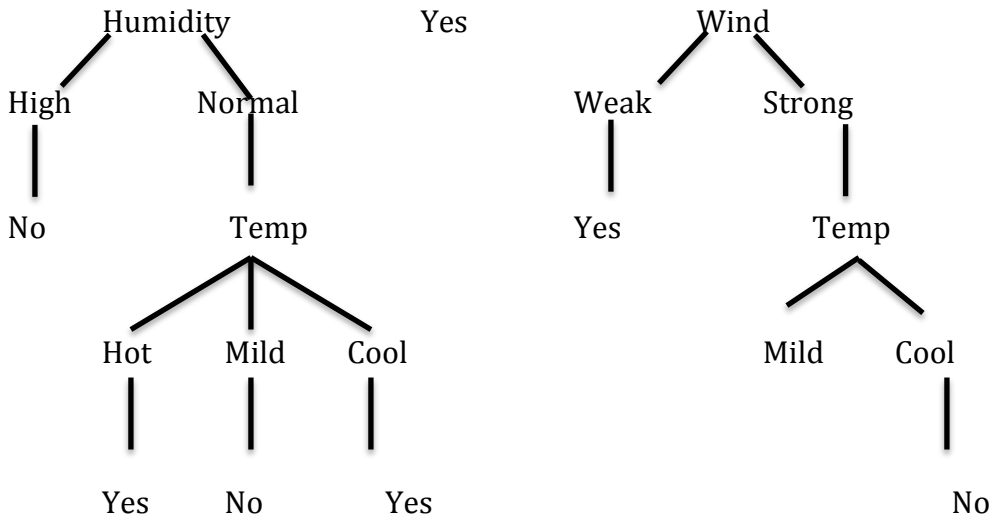
We need to split on two branches.

		Sunny - Normal			
		Yes	No		Entropy
Temp	Hot	1	0	1	0
	Mild	0	1	1	0
	Cool	1	0	1	0
				3	

We do not need to calculate the Information Gain. They are all leaves.

		Rain - Strong			
		Yes	No		Entropy
Temp	Hot	0	0	0	0
	Mild	1	1	2	0
	Cool	0	1	1	0
				3	



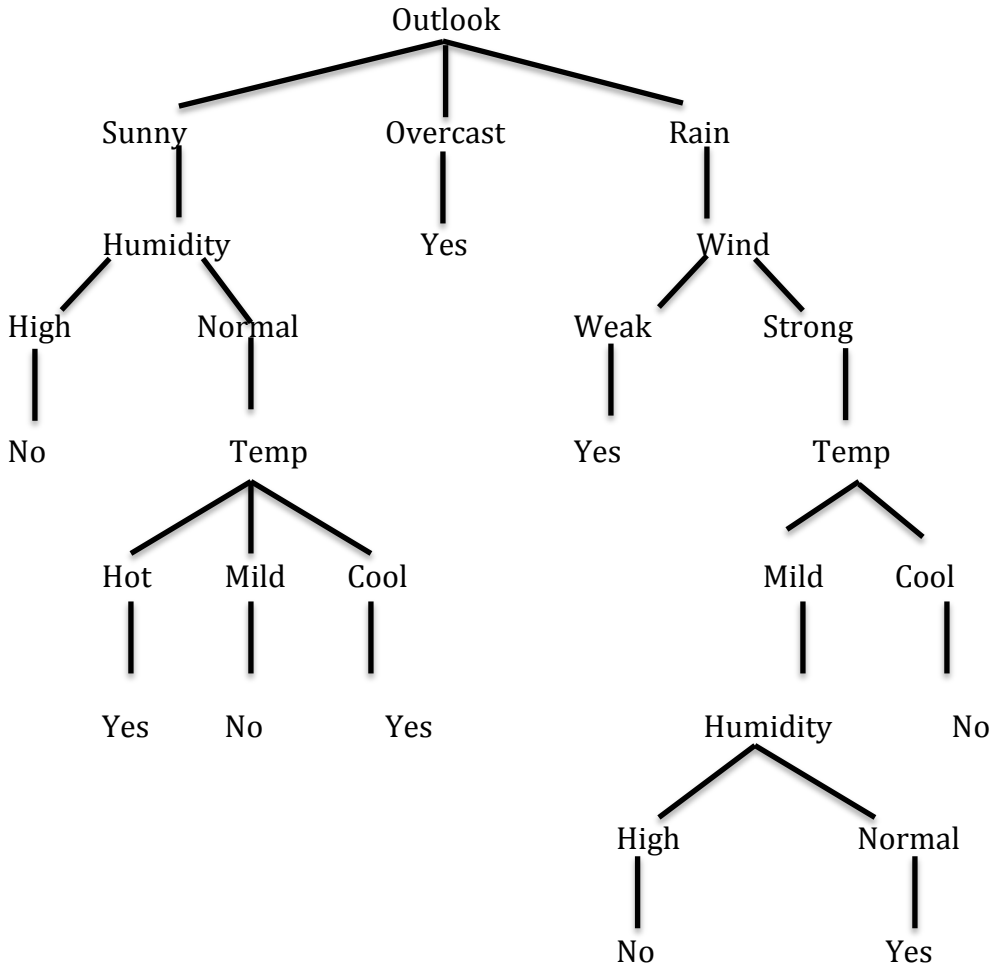


Level 4:

We need to go on until reaching a leaf.

		Rain – Strong - Mild			
		Yes	No		Entropy
Humidity	High	0	1	1	0
	Normal	1	0	1	0
				2	

Complete tree:



Equivalent rule set:

$$IF\ Outlook = Sunny\ \wedge Humidity = High\ THEN\ Play = No$$

$$IF\ Outlook = Sunny\ \wedge Humidity = Normal\ \wedge Temp = Hot\ THEN\ Play = Yes$$

$$IF\ Outlook = Sunny\ \wedge Humidity = Normal\ \wedge Temp = Mild\ THEN\ Play = No$$

$$IF\ Outlook = Sunny\ \wedge Humidity = Normal\ \wedge Temp = Cool\ THEN\ Play = Yes$$

$$IF\ Outlook = Overcast\ THEN\ Play = Yes$$

$$IF\ Outlook = Rain\ \wedge Wind = Weak\ \ THEN\ Play = Yes$$

$$IF\ Outlook = Rain\ \wedge Wind = Strong\ \wedge Temp = Mild\ \wedge Humidity = High\ THEN\ Play = No$$

$$IF\ Outlook = Rain\ \wedge Wind = Strong\ \wedge Temp = Mild\ \wedge Humidity = Normal\ THEN\ Play = Yes$$

$$IF\ Outlook = Rain\ \wedge Wind = Strong\ \wedge Temp = Cool\ THEN\ Play = No$$

Question 4:

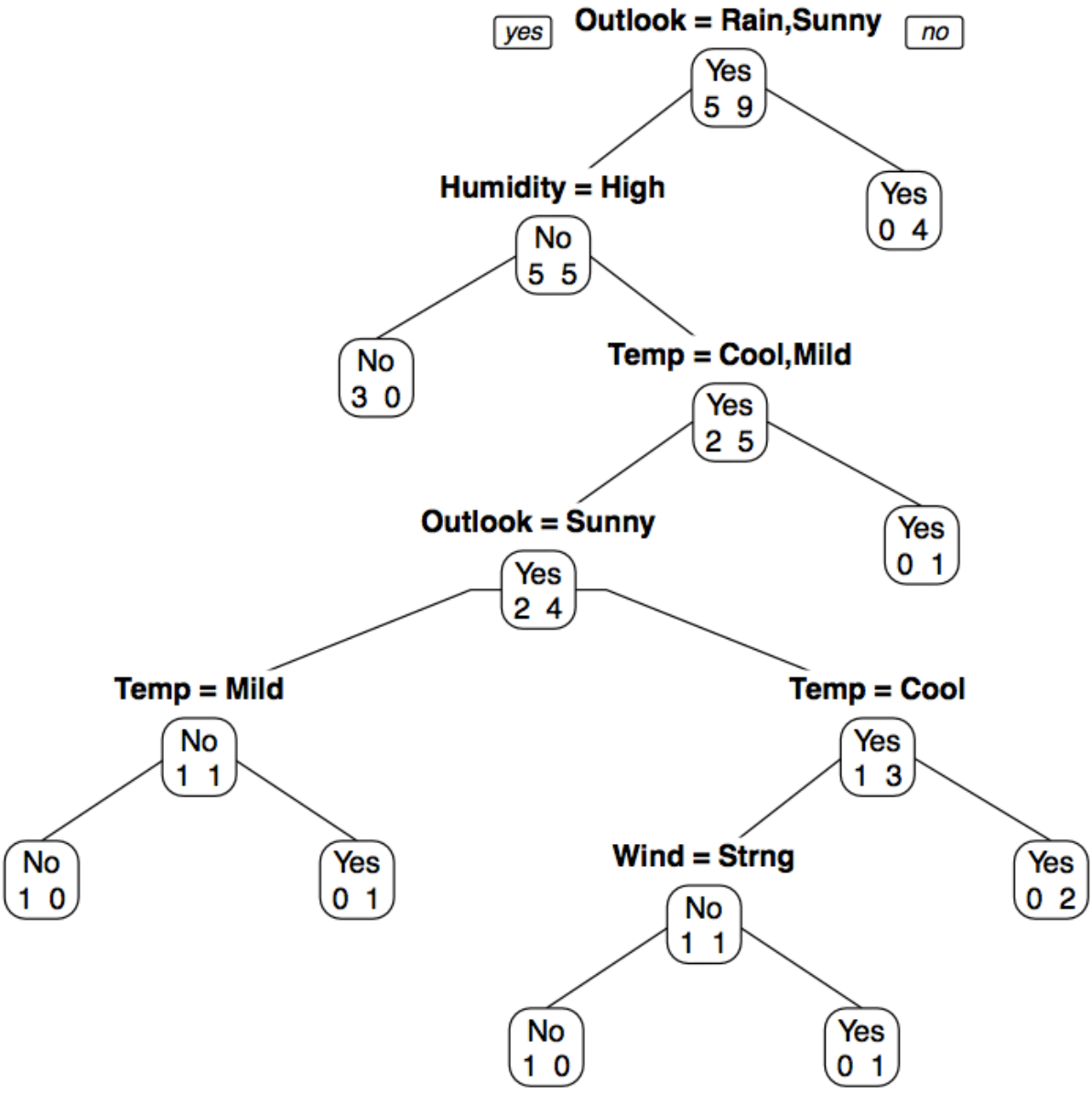
Explain the significance of the argument control=rpart.control(minsplit=x) on the behavior of rpart().
Minsplit : "the minimum number of observations that must exist in a node in order for a split to be attempted" is the definition of this argument given by the R documentation.

The argument will change the number of computation needed to construct the tree. Indeed, if we set up this argument low, this will mean that we allow splitting more easily on an attribute as we allow an attribute with a few instances to be considered and be potentially a node. If we setup this argument higher, w are less tolerant with we demand a higher number of instances in order to consider the attribute and be potentially a node. In this case, the number of computation would decrease.
Furthermore, if an attribute has fewer instances than the number required by this argument, this means that we consider that this attribute is irrelevant and can reduce of the accuracy of the decision tree performance. So, this is a pre-pruning method as it would not split if the sub-tree contain fewer instances than the threshold number of instances (minsplit).

fit <- rpart(Play ~ Outlook + Temp + Humidity + Wind, data = play,
parms=list(split='information'), control=rpart.control(minsplit=2))

I chose minsplit = 2 in order to maximize the number of splits and to expand each branch until the complete extent of the tree.

prp(fit, type=1, extra=1, faclen=5)



Decision Tree generated

Differences:

- The degree of this tree is 5, whereas the one generated manually is 4.
- The rule set is different for the other generated manually.
- The leaves of this tree can either be yes or no for a split. Which means that for example, for Wind=Strong on the lowest level, we can either have outputs classified by yes or by no. However, within the manually generated one, each leaf ends up by classifying the instance. It seems like the rules are more generals for this tree and more specific on the manually generated one.