Machine Learning with Python

MTH786U/P 2023/24

Week 12: Decision Trees

Nicola Perra, Queen Mary University of London (QMUL)

n.perra@qmul.ac.uk

How can we deal with data like this one?

Shadow	Garlic	Complexion	Accent	Vampire
?	Yes	Pale	None	No
Yes	Yes	Ruddy	None	No
?	No	Ruddy	None	Yes
No	No	Average	Heavy	Yes
?	No	Average	Odd	Yes
Yes	No	Pale	Heavy	No
Yes	No	Average	Heavy	No
?	Yes	Ruddy	Odd	No

Issues

All data is categorical

Issues

All data is categorical

Some features might not matter

Issues

All data is categorical

Some features might not matter

Some features might be important only is some cases

Issues

All data is categorical

Some features might not matter

Some features might be important only is some cases

Cost associated to different “measurements” to assess different features might not be the same

Identification trees

Identification trees offer a solution to these problems

Identification trees

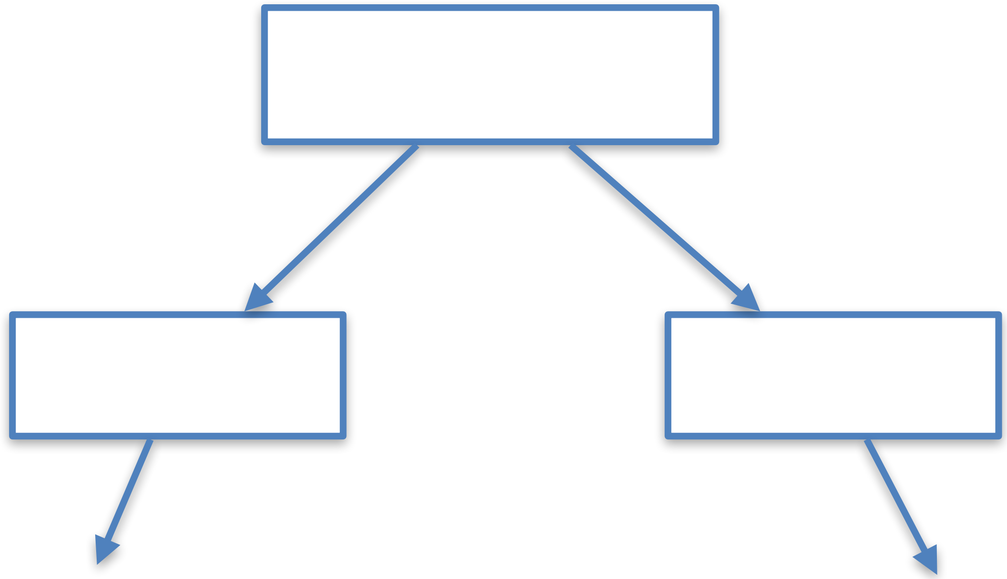
Identification trees offer a solution to these problems

The intuition is that you can stack the most relevant tests and classify the data progressively

Identification trees

Identification trees offer a solution to these problems

The intuition is that you can stack the most relevant tests and classify the data progressively



Identification trees

What are the characteristics of a good identification tree?

Identification trees

What are the characteristics of a good identification tree?

Classify the data well with a small number of “tests” (called stumps)

Identification trees

What are the characteristics of a good identification tree?

Classify the data well with a small number of “tests” (called stumps)

The longer the tree the higher the probability of overfitting

Identification trees

What are the characteristics of a good identification tree?

Classify the data well with a small number of “tests” (called stumps)

The longer the tree the higher the probability of overfitting

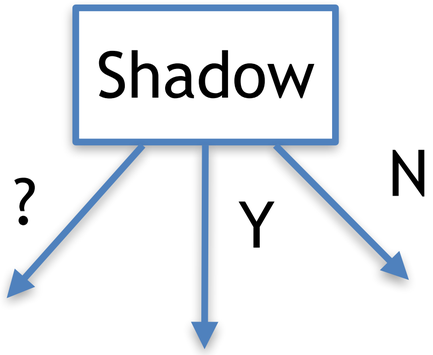
How do we select the number and order of tests to do?!

Identification trees

Let see how each test perform at the first level

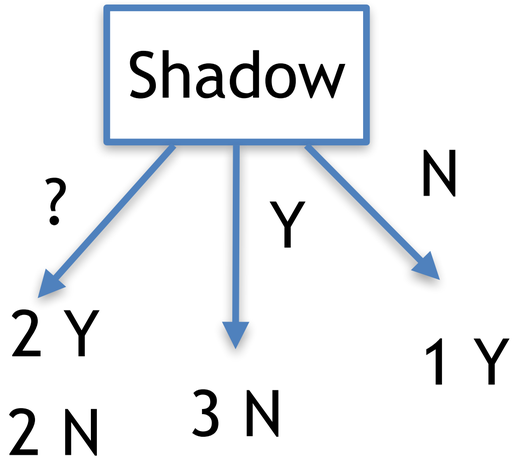
Identification trees

Let see how each test perform at the first level



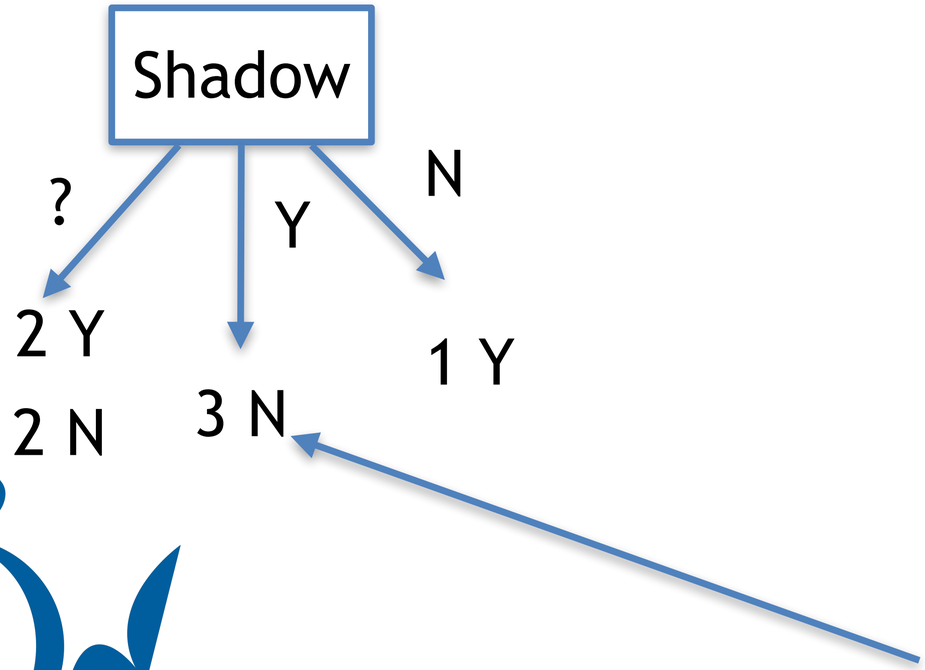
Identification trees

Let see how each test perform at the first level



Identification trees

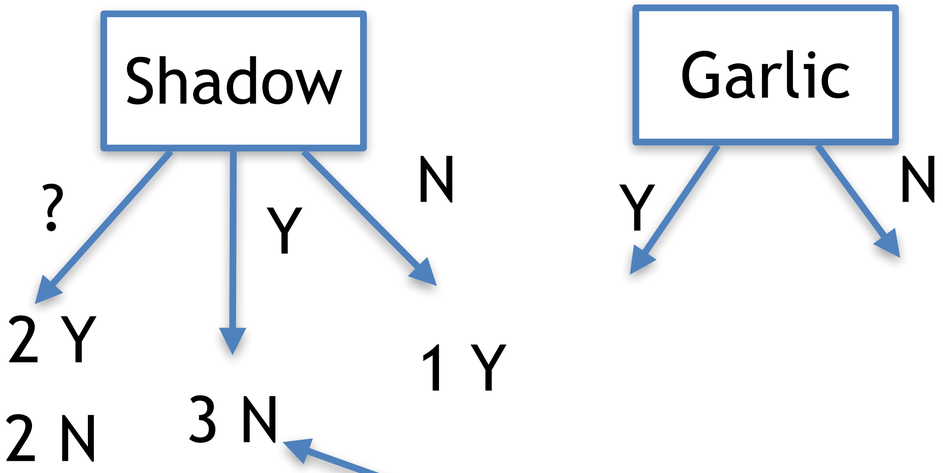
Let see how each test perform at the first level



If they have a shadow they appear not to be vampires.

Identification trees

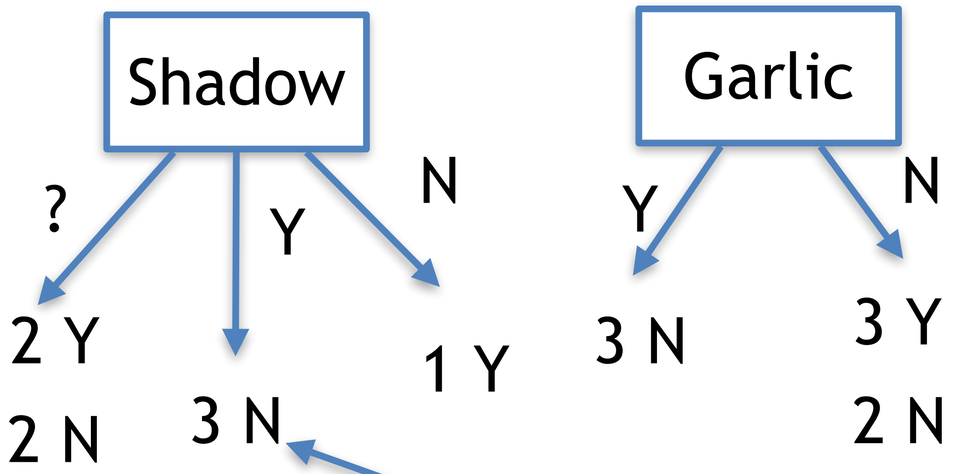
Let see how each test perform at the first level



If they have a shadow they appear not to be vampires.

Identification trees

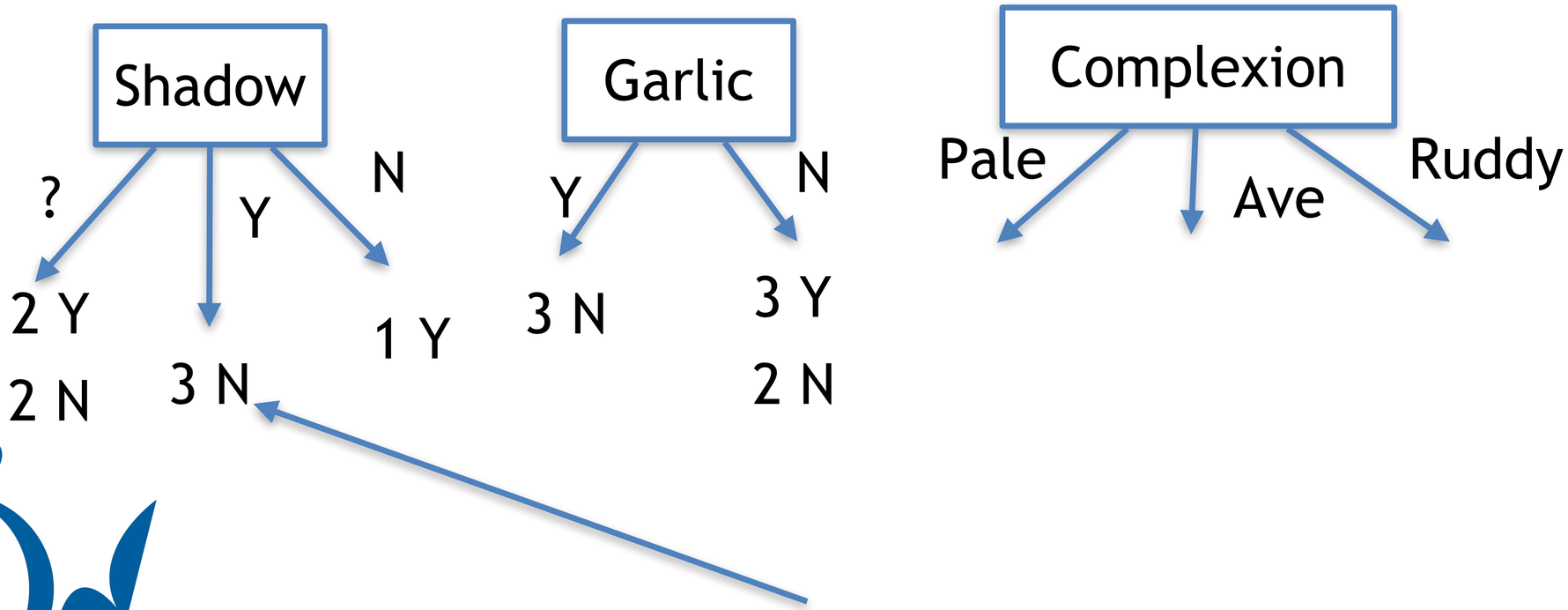
Let see how each test perform at the first level



If they have a shadow they appear not to be vampires.

Identification trees

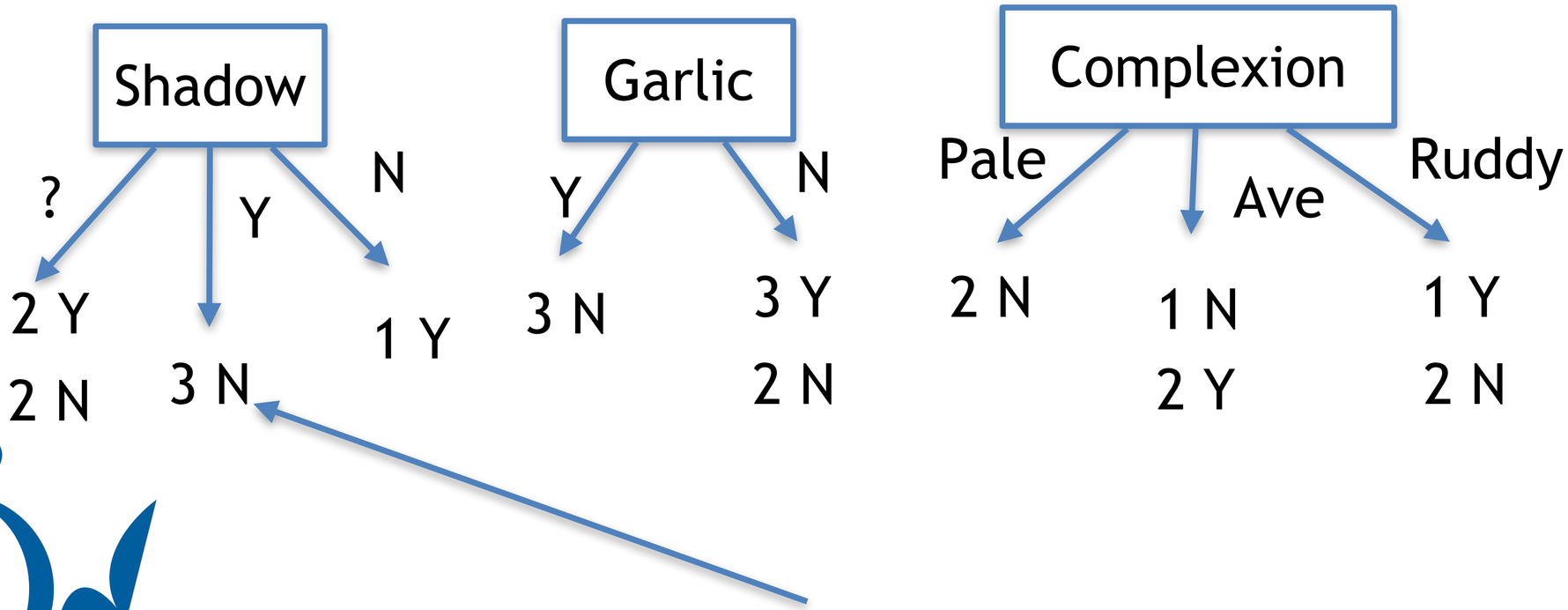
Let see how each test perform at the first level



If they have a shadow they appear not to be vampires.

Identification trees

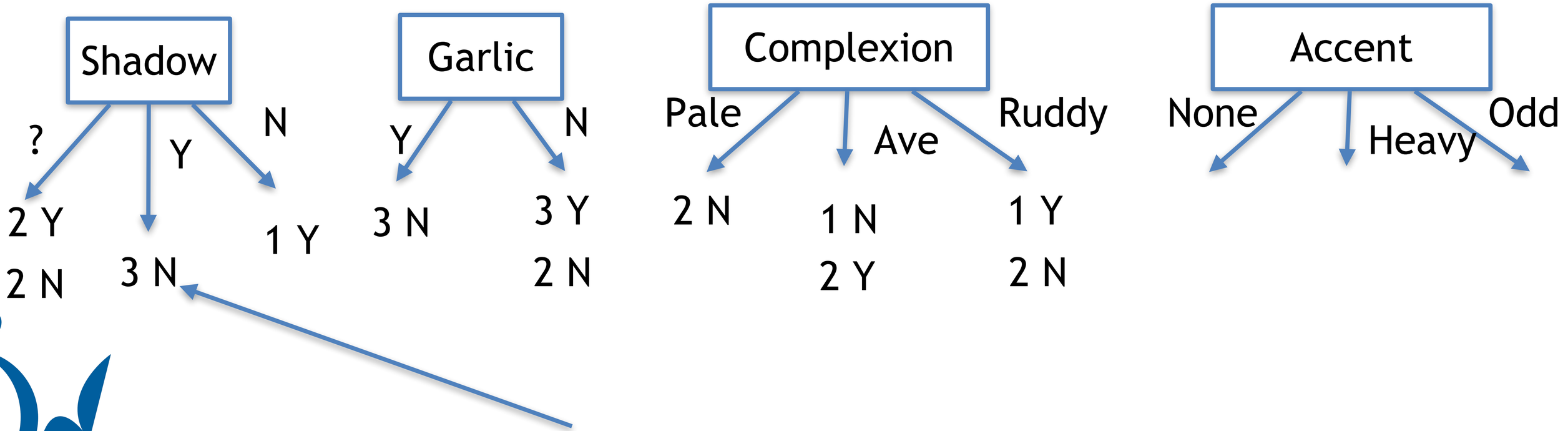
Let see how each test perform at the first level



If they have a shadow they appear not to be vampires.

Identification trees

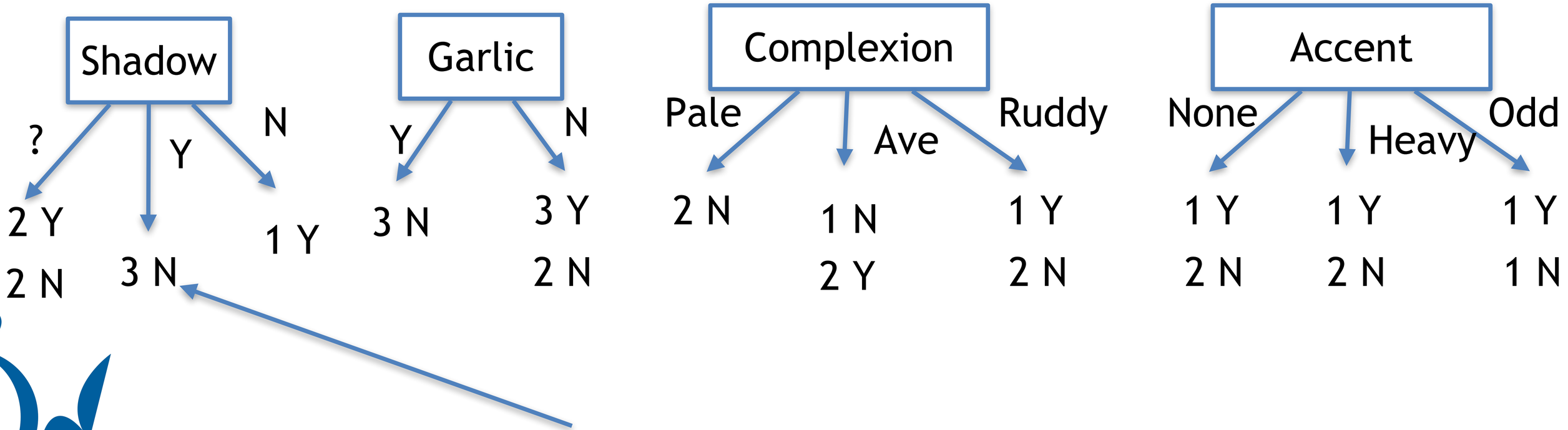
Let see how each test perform at the first level



If they have a shadow they appear not to be vampires.

Identification trees

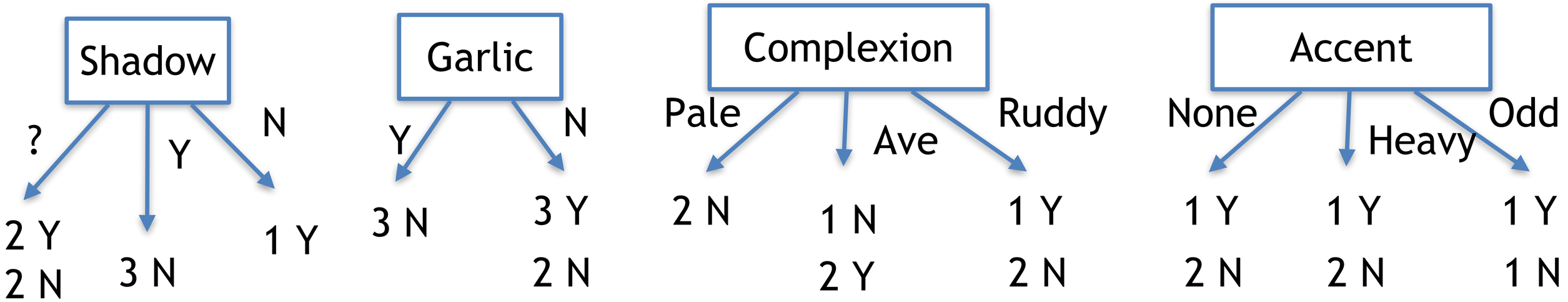
Let see how each test perform at the first level



If they have a shadow they appear not to be vampires.

Identification trees

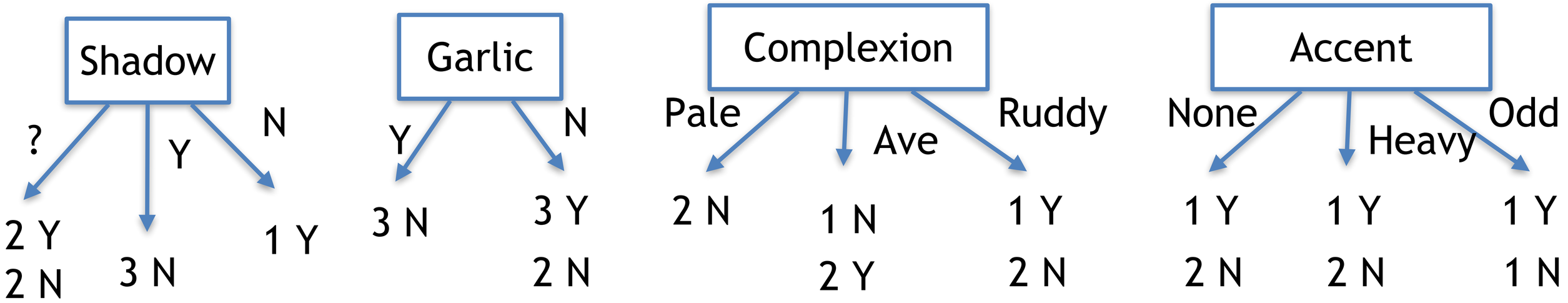
Which is the best to use?



We should look for tests that divide the sample in homogenous samples

Identification trees

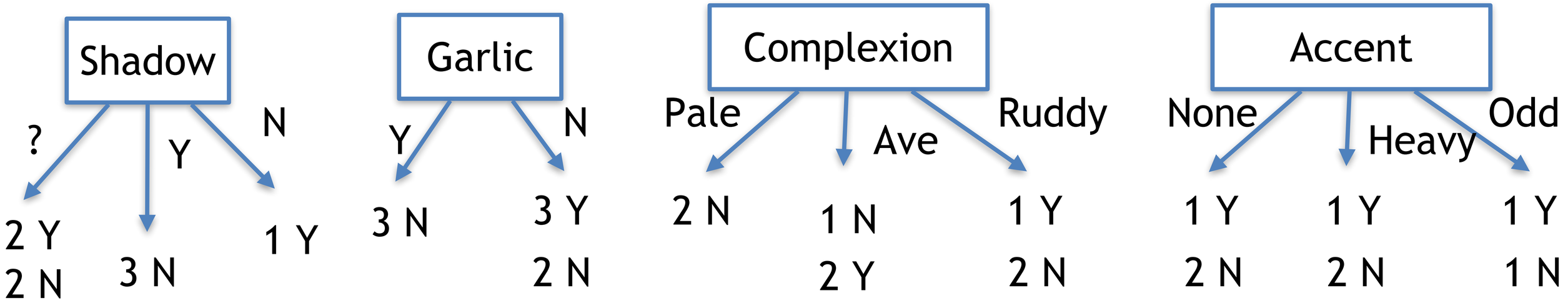
Which is the best to use?



Idea: let us use a “score” of each test the number of samples that are split in homogenous classes

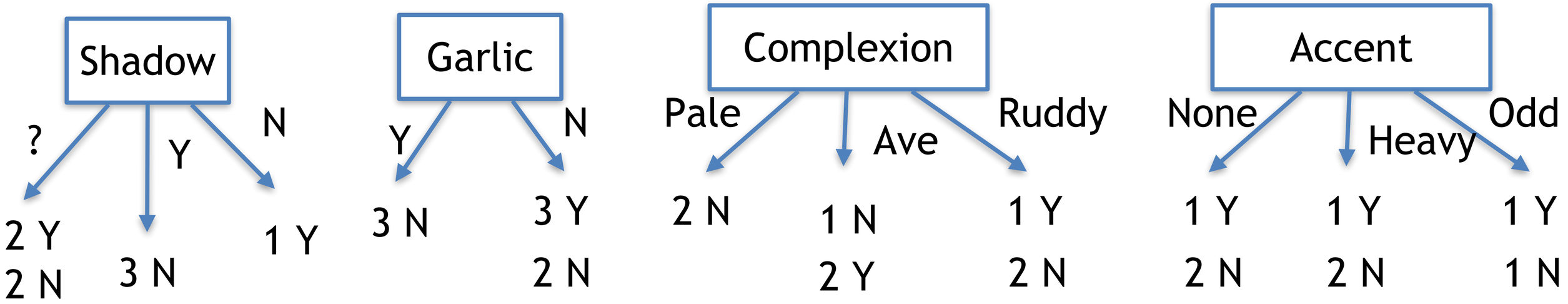
Identification trees

Which is the best to use?



Identification trees

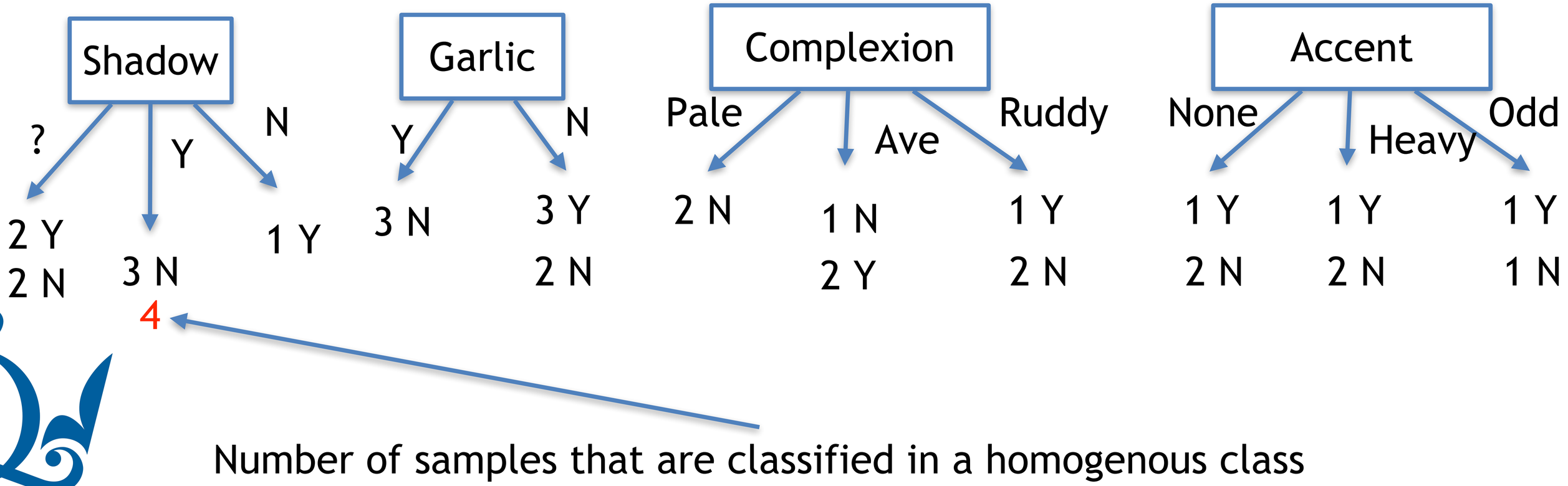
Which is the best to use?



Number of samples that are classified in a homogenous class

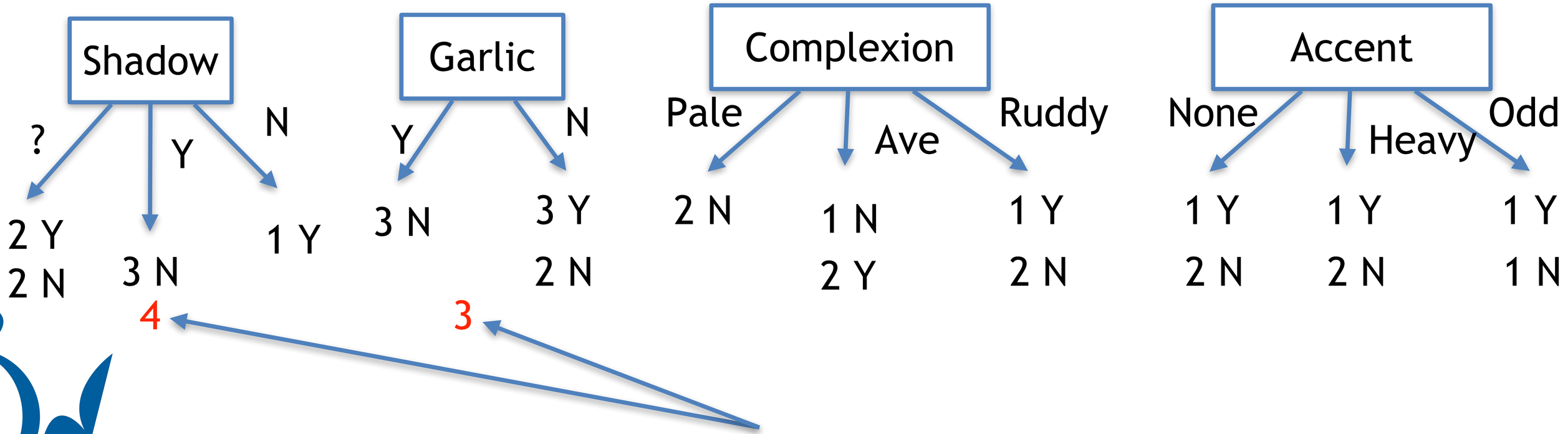
Identification trees

Which is the best to use?



Identification trees

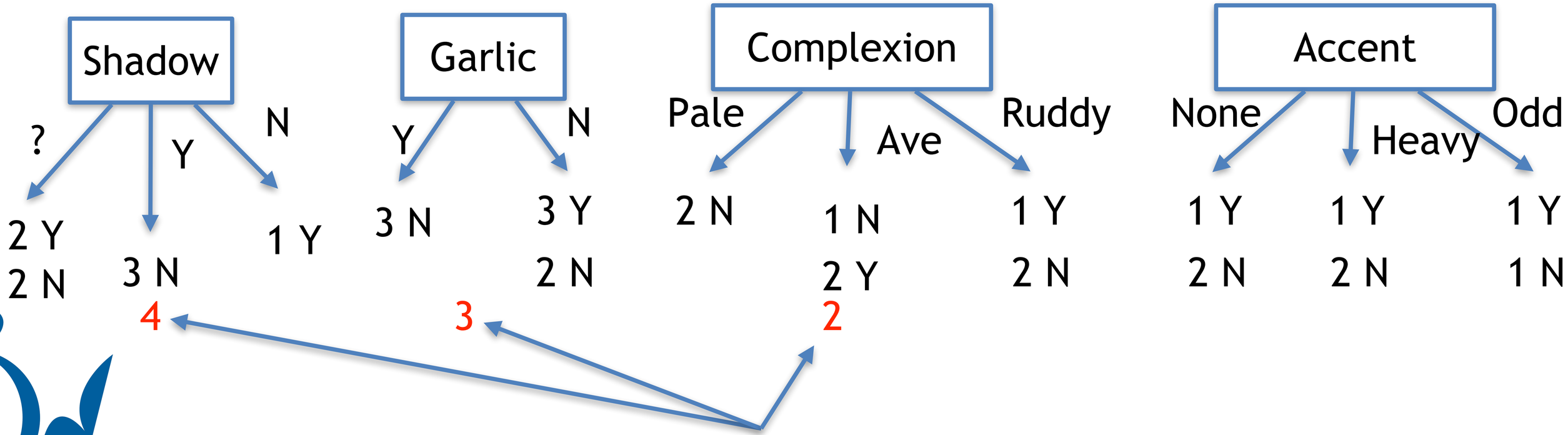
Which is the best to use?



Number of samples that are classified in a homogenous class

Identification trees

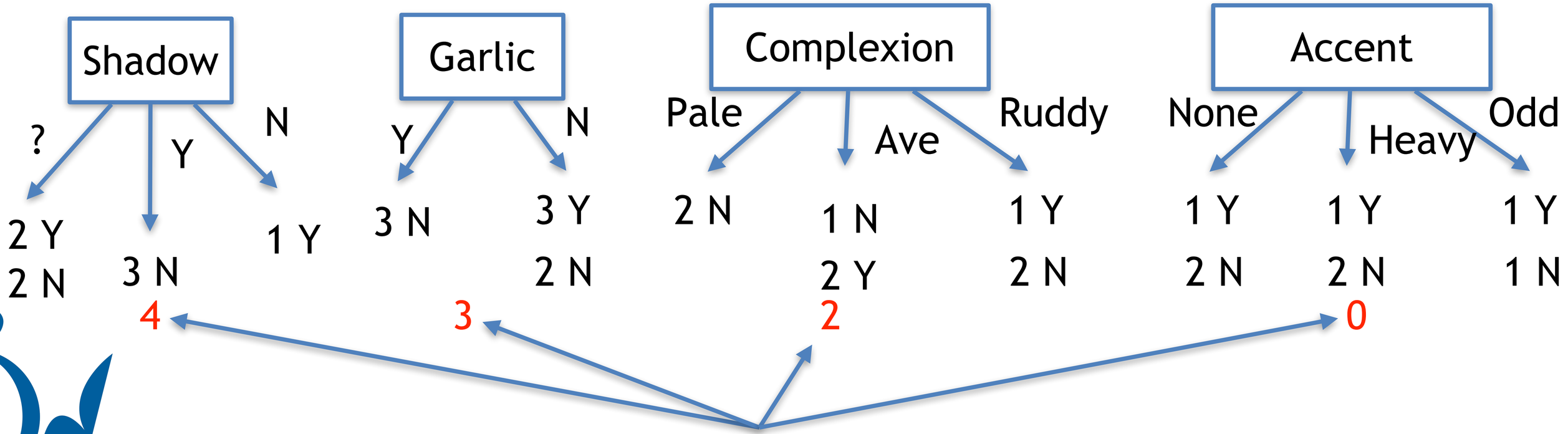
Which is the best to use?



Number of samples that are classified in a homogenous class

Identification trees

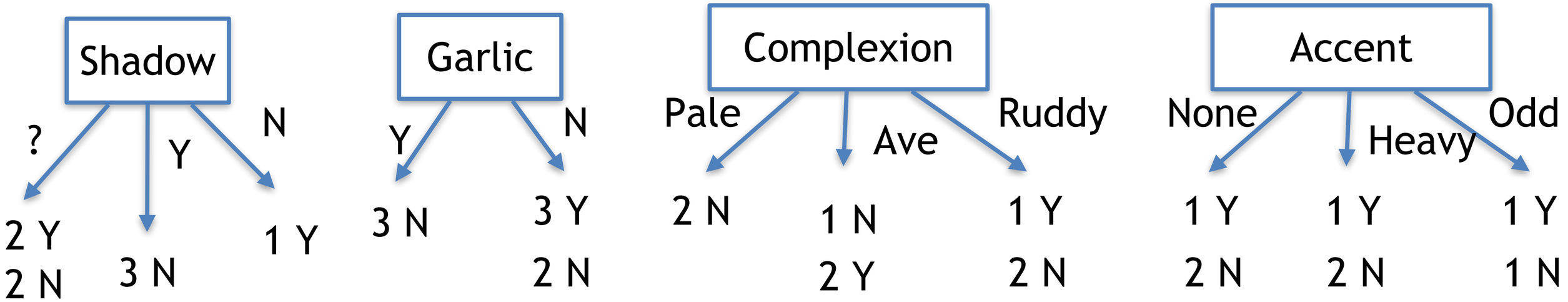
Which is the best to use?



Number of samples that are classified in a homogenous class

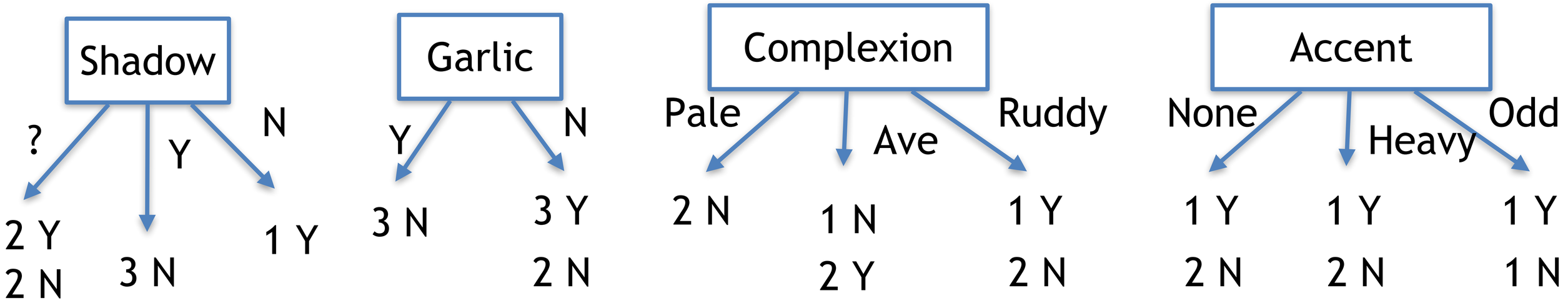
Identification trees

Shadow appears to be the best according to this score



Identification trees

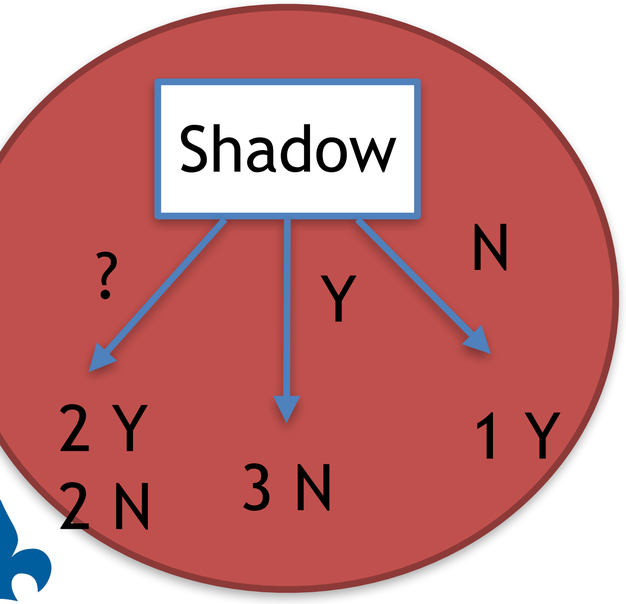
Shadow appears to be the best according to this score

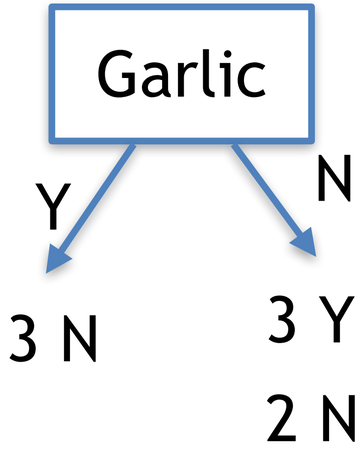


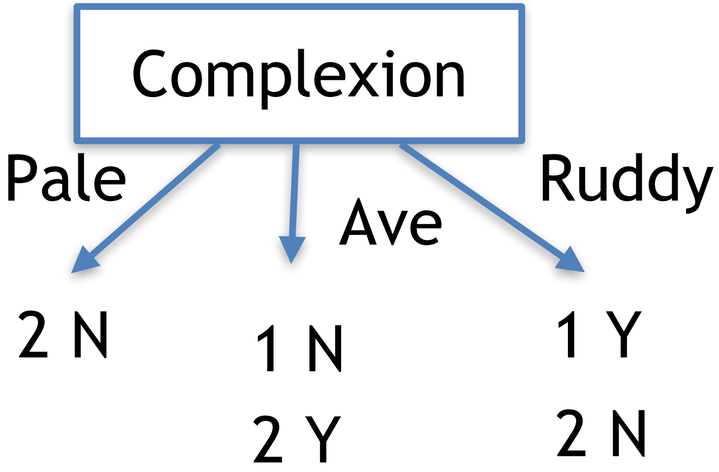
We can then select it as the root of the tree

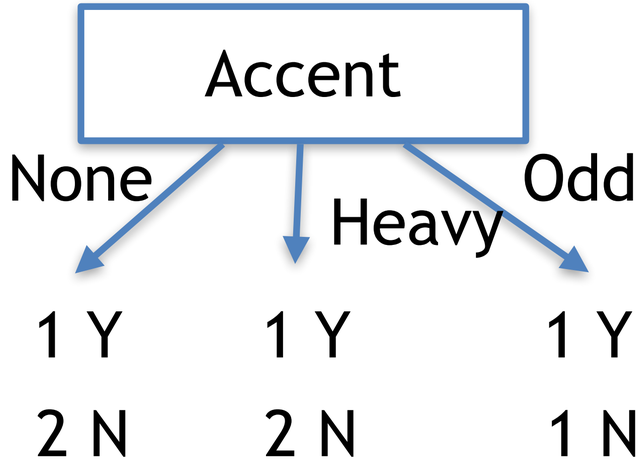
Identification trees

Shadow appears to be the best according to this score



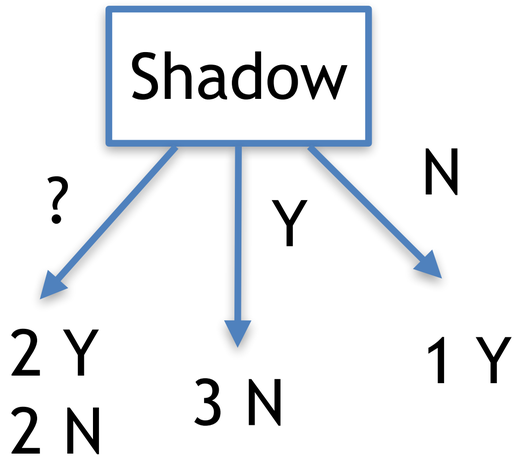




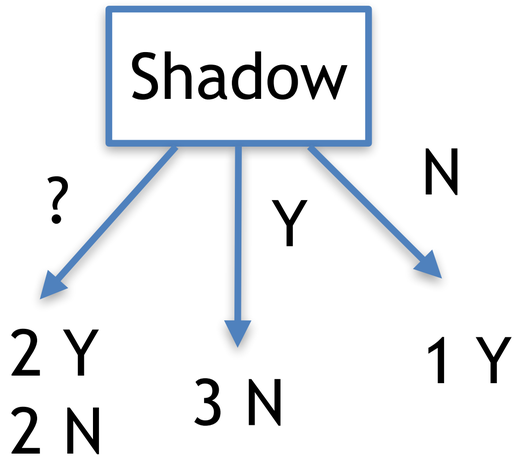


We can then select it as the root of the tree

Identification trees

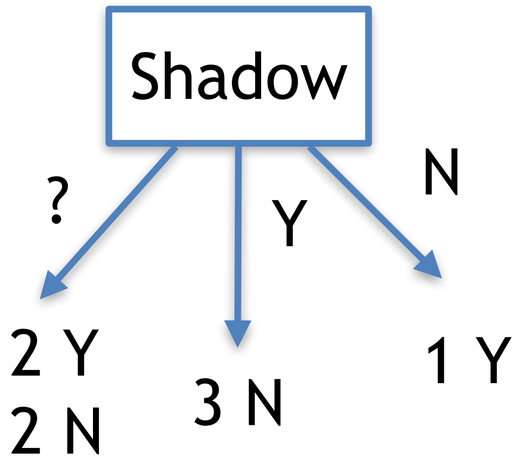


Identification trees



We can apply the same method to the branch that is not homogenous

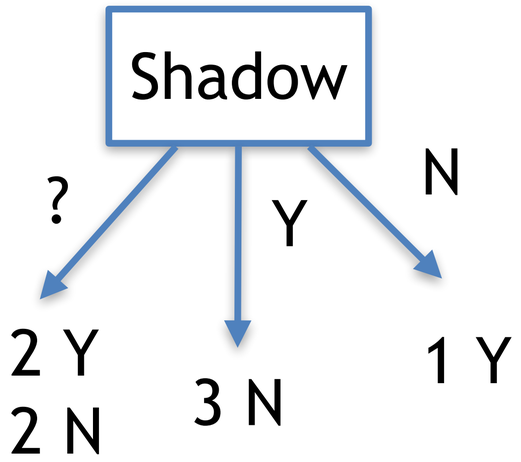
Identification trees



We can apply the same method to the branch that is not homogenous

Shadow	Garlic	Complexion	Accent	Vampire
?	Yes	Pale	None	No
?	No	Ruddy	None	Yes
?	No	Average	Odd	Yes
?	Yes	Ruddy	Odd	No

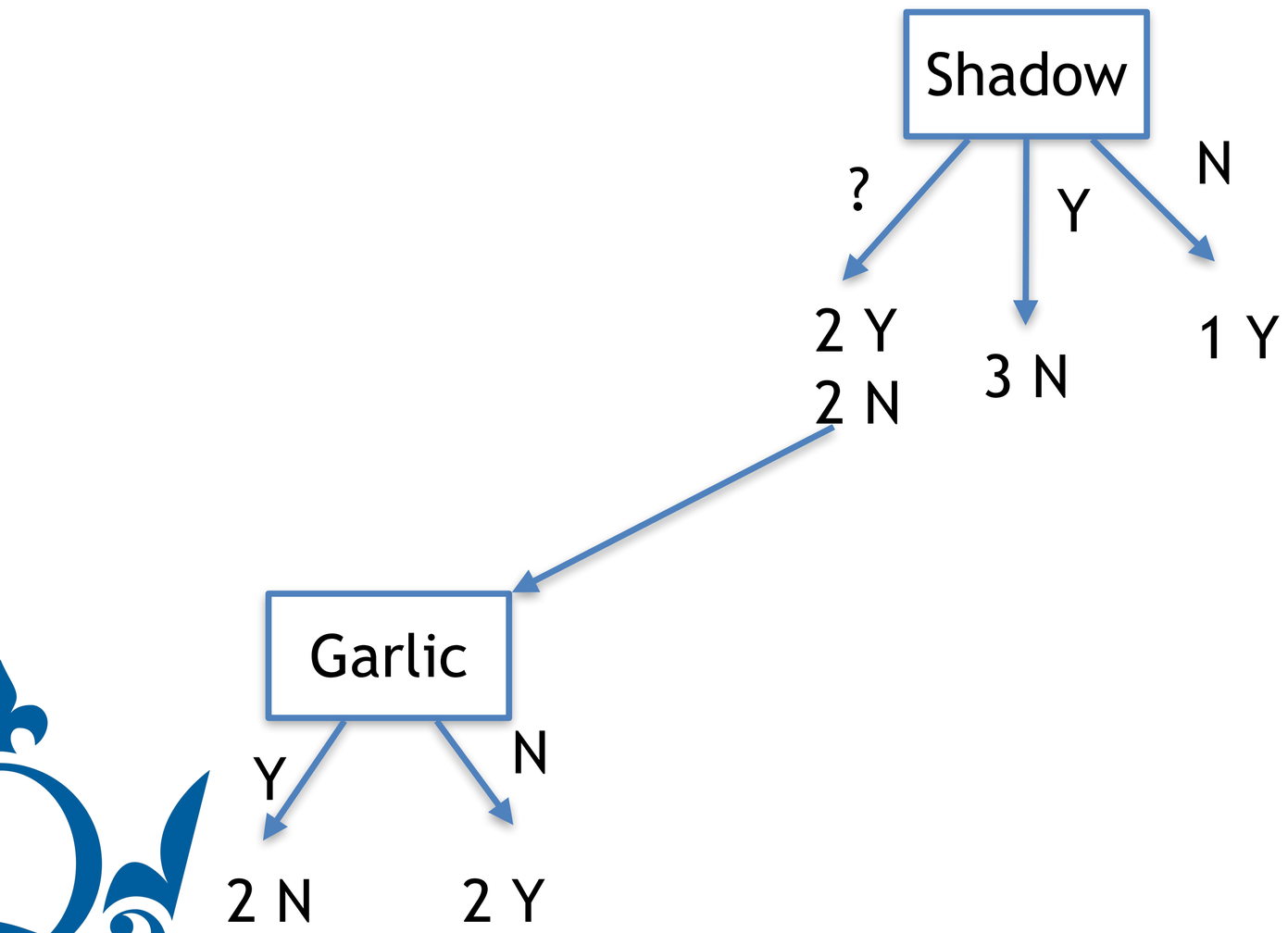
Identification trees



Remaining data

Shadow	Garlic	Complexion	Accent	Vampire
?	Yes	Pale	None	No
?	No	Ruddy	None	Yes
?	No	Average	Odd	Yes
?	Yes	Ruddy	Odd	No

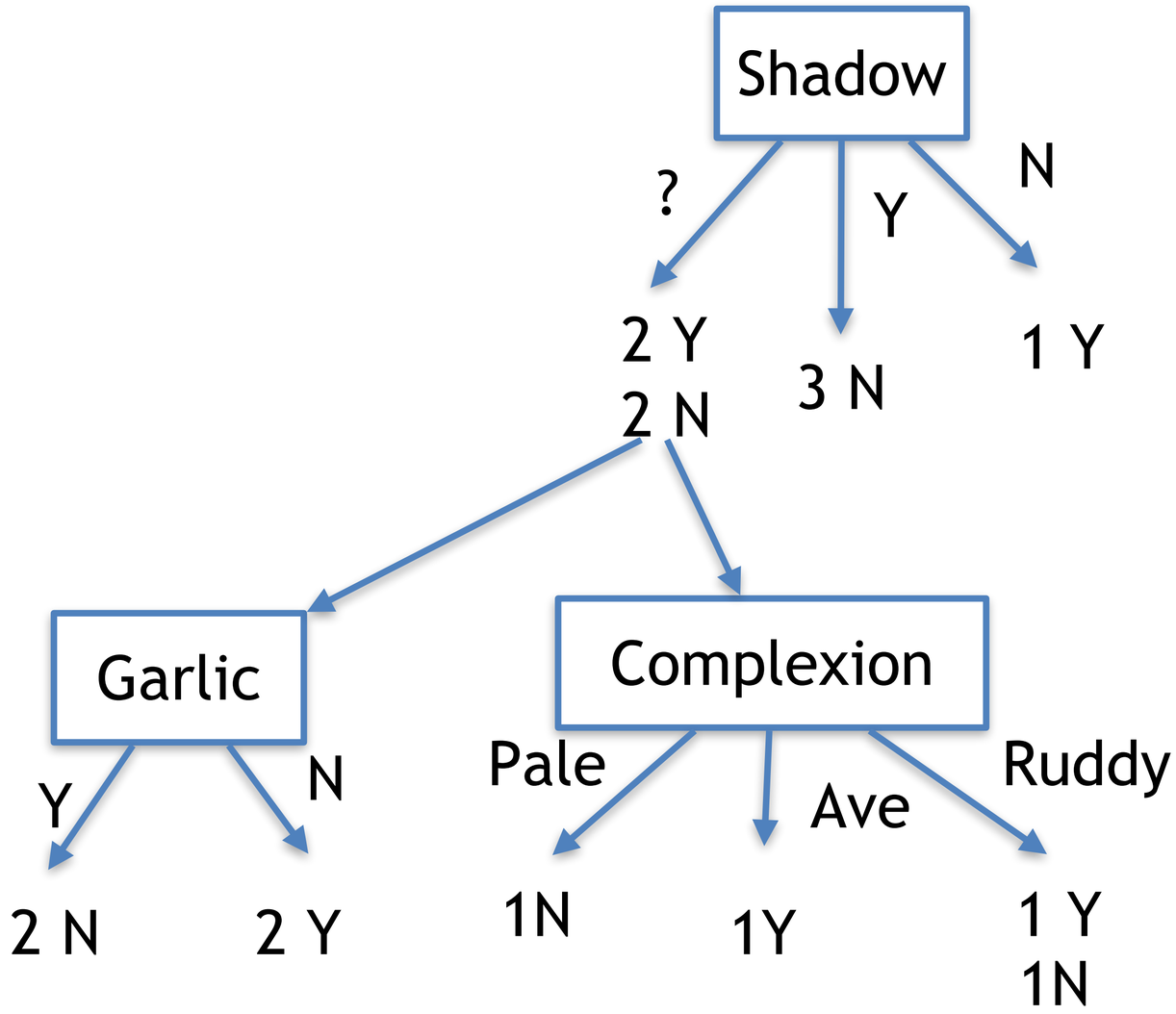
Identification trees



Remaining data

Shadow	Garlic	Complexion	Accent	Vampire
?	Yes	Pale	None	No
?	No	Ruddy	None	Yes
?	No	Average	Odd	Yes
?	Yes	Ruddy	Odd	No

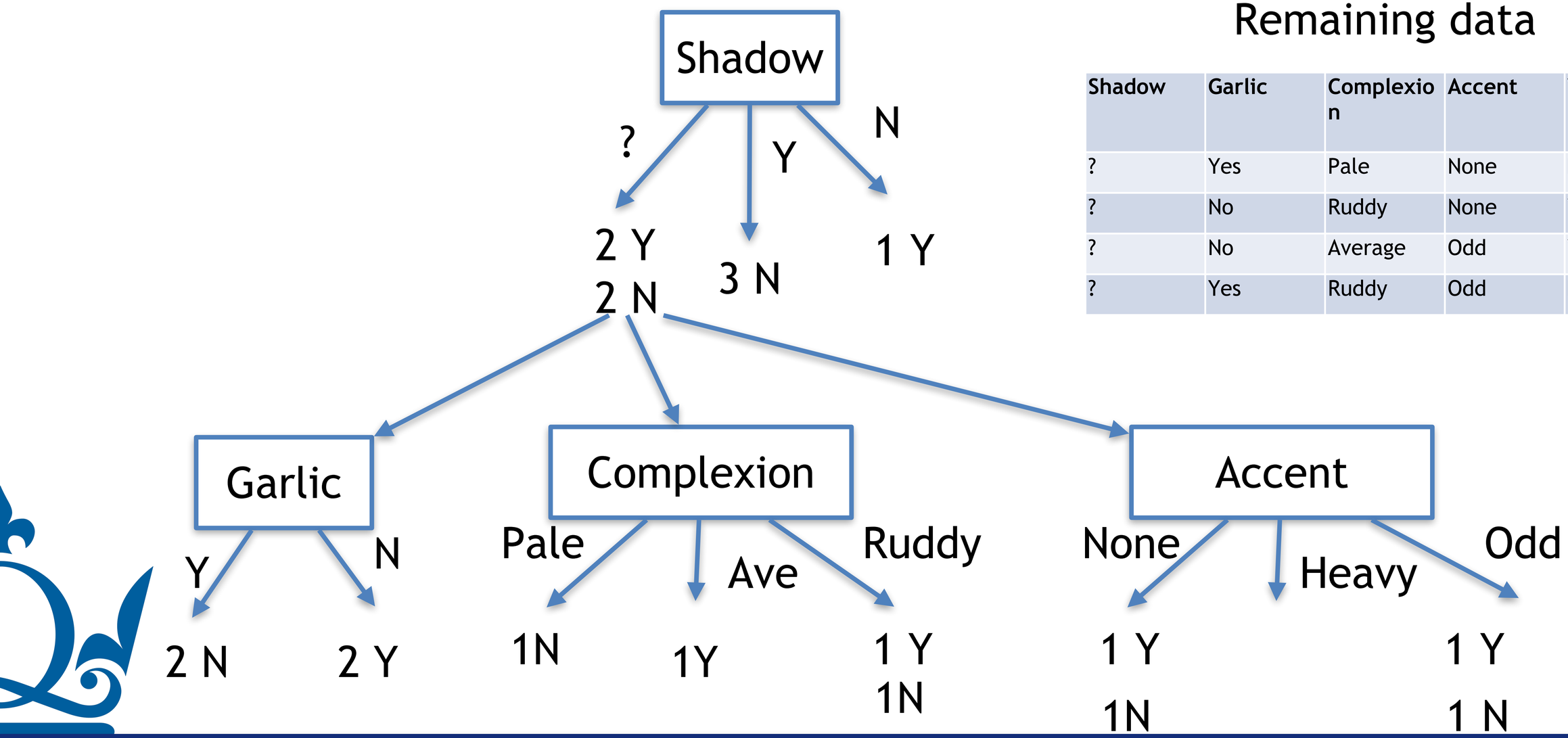
Identification trees



Remaining data

Shadow	Garlic	Complexion	Accent	Vampire
?	Yes	Pale	None	No
?	No	Ruddy	None	Yes
?	No	Average	Odd	Yes
?	Yes	Ruddy	Odd	No

Identification trees

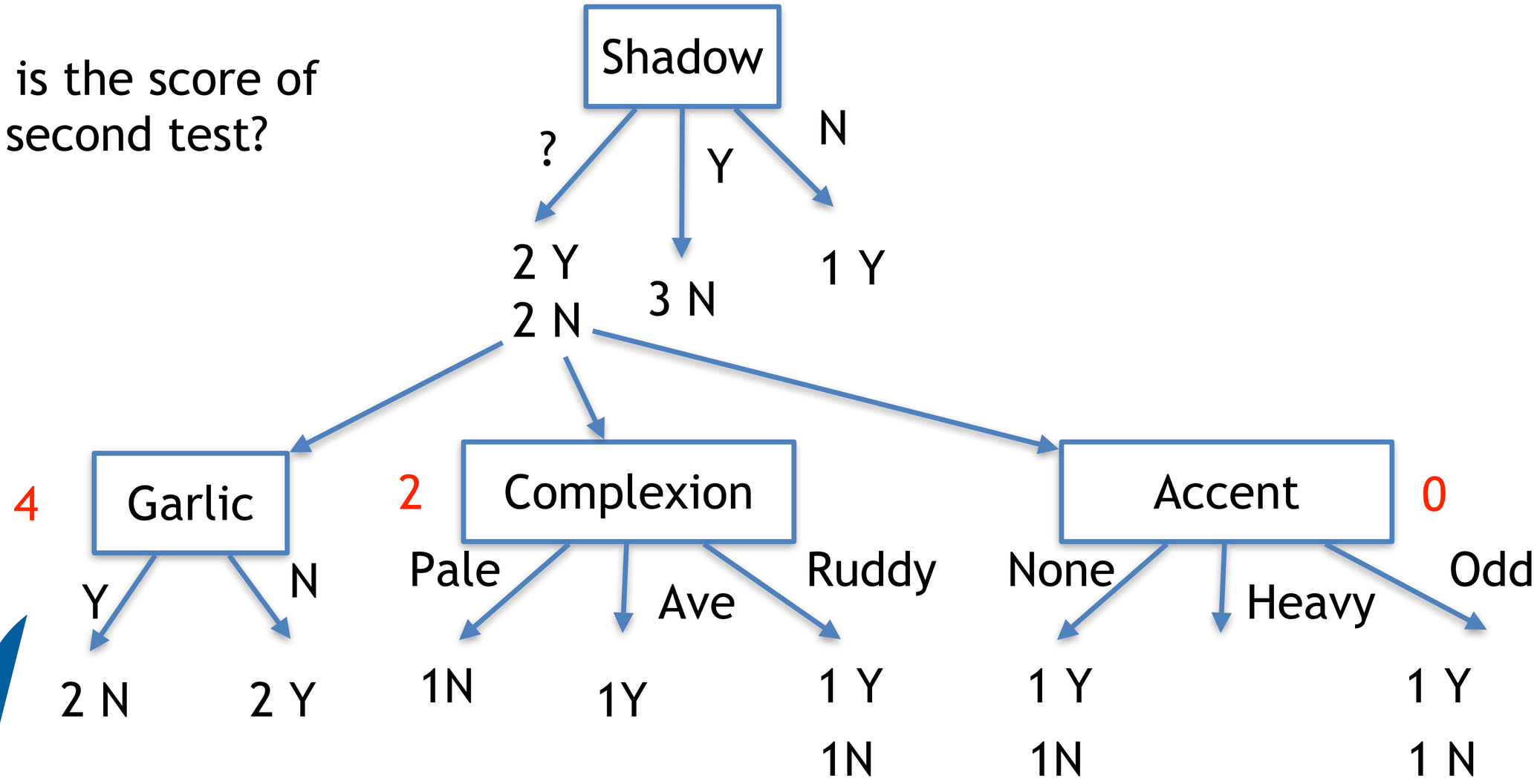


Remaining data

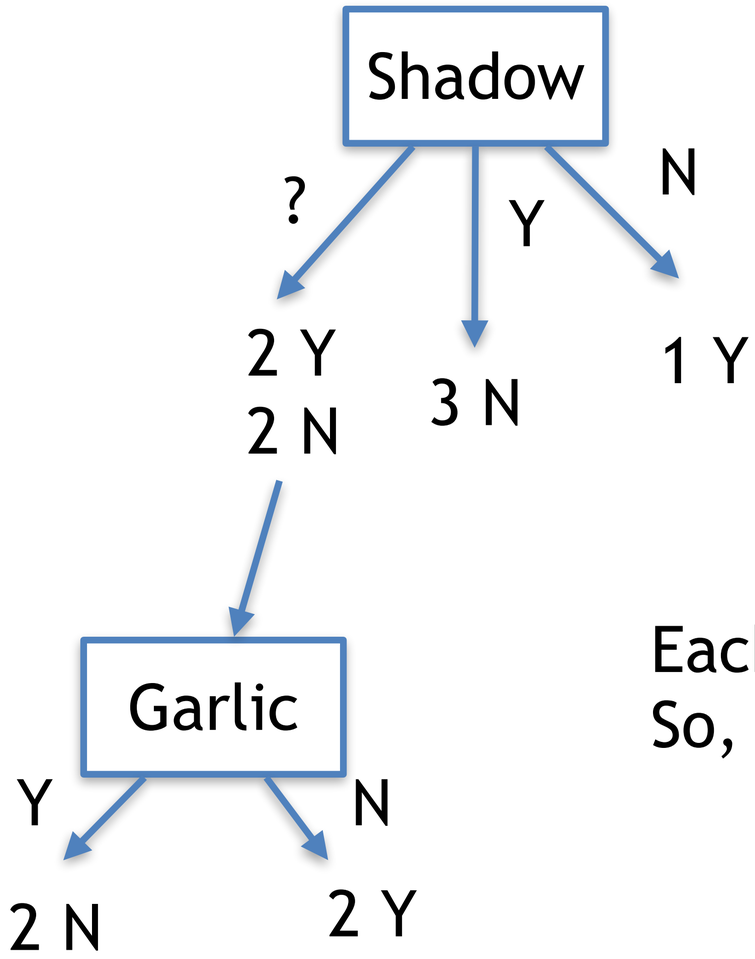
Shadow	Garlic	Complexion	Accent	Vampire
?	Yes	Pale	None	No
?	No	Ruddy	None	Yes
?	No	Average	Odd	Yes
?	Yes	Ruddy	Odd	No

Identification trees

What is the score of each second test?



Identification trees



Each sample is now in a homogenous class!
So, we are done!

Identification trees

In real application it is rare to find perfectly homogenous classes

Identification trees

In real application it is rare to find perfectly homogenous classes

What can we do? We need a less strict scoring system

Identification trees

In real application it is rare to find perfectly homogenous classes

What can we do? We need a less strict scoring system

Idea: perfect homogenous branches are great, but we can a measure of disorder more in general

Quantifying disorder

We can use metrics from information theory

$$D(set) = -\frac{P}{T}\log_2\left(\frac{P}{T}\right) - \frac{N}{T}\log_2\left(\frac{N}{T}\right)$$

Quantifying disorder

We can use metrics from information theory

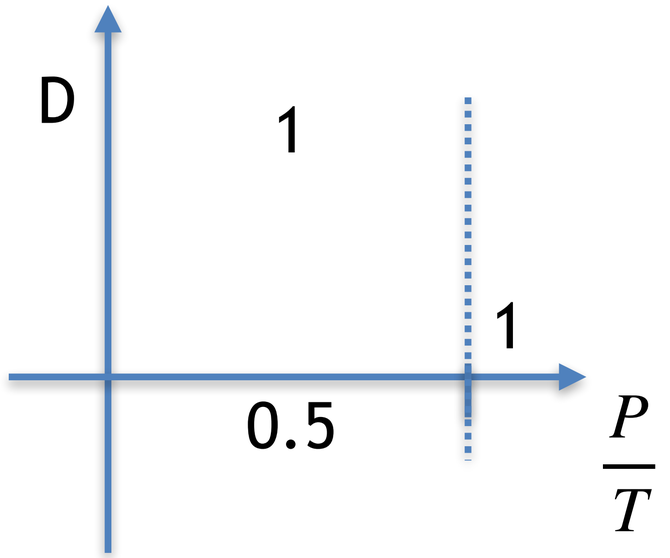
$$D(set) = -\frac{P}{T}\log_2\left(\frac{P}{T}\right) - \frac{N}{T}\log_2\left(\frac{N}{T}\right)$$

Where D is the disorder, P are the positive samples, N the negative samples, and T the total number of samples $T = P + N$

Quantifying disorder

How does D looks like as function of P/T?

$$D(set) = -\frac{P}{T}\log_2\left(\frac{P}{T}\right) - \frac{N}{T}\log_2\left(\frac{N}{T}\right)$$

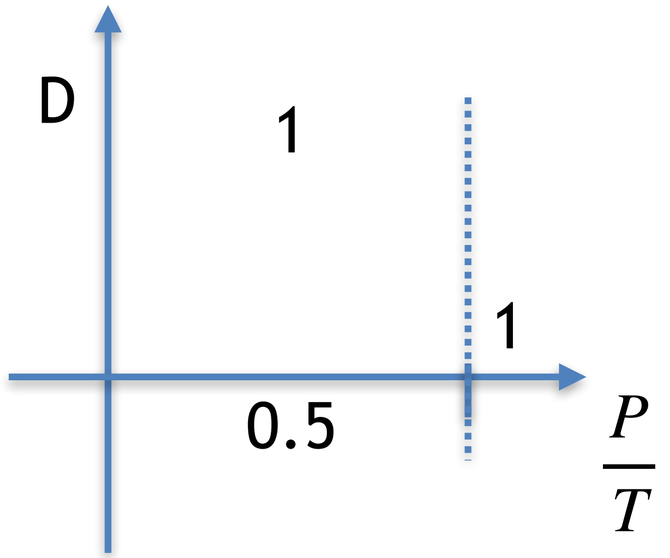


Quantifying disorder

How does D looks like as function of P/T?

$$D(set) = -\frac{P}{T}\log_2\left(\frac{P}{T}\right) - \frac{N}{T}\log_2\left(\frac{N}{T}\right)$$

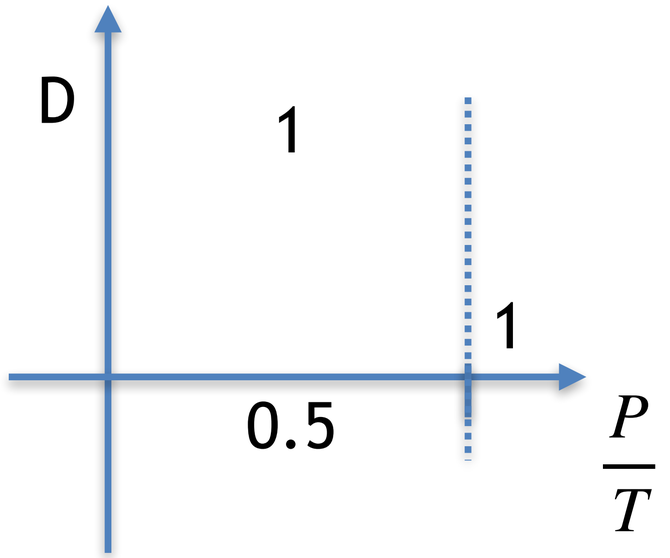
If P=N?



Quantifying disorder

How does D looks like as function of P/T?

$$D(set) = -\frac{P}{T}\log_2\left(\frac{P}{T}\right) - \frac{N}{T}\log_2\left(\frac{N}{T}\right)$$



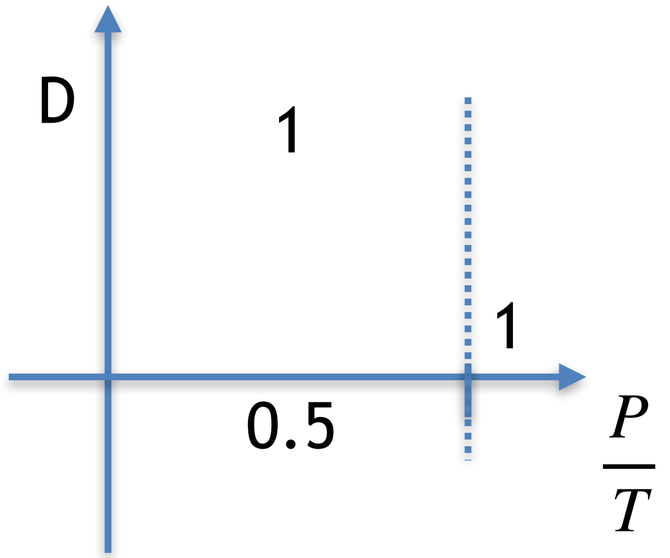
If P=N?

$$D(set) = -\frac{1}{2}\log_2\left(\frac{1}{2}\right) - \frac{1}{2}\log_2\left(\frac{1}{2}\right)$$

Quantifying disorder

How does D looks like as function of P/T?

$$D(set) = -\frac{P}{T}\log_2\left(\frac{P}{T}\right) - \frac{N}{T}\log_2\left(\frac{N}{T}\right)$$



If P=N?

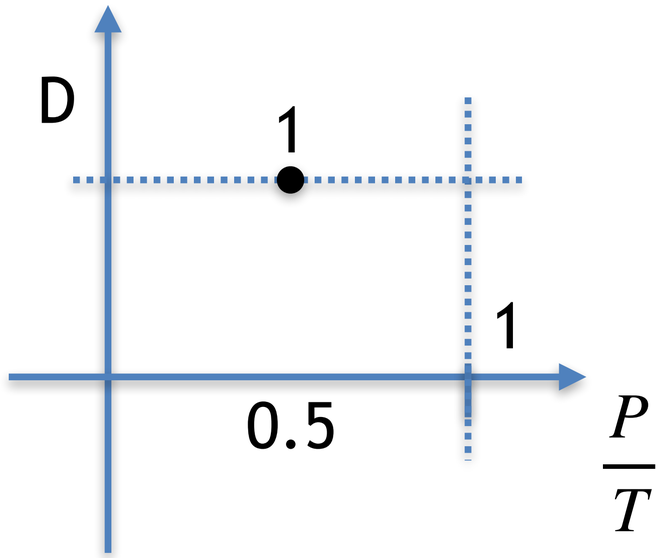
$$D(set) = -\frac{1}{2}\log_2\left(\frac{1}{2}\right) - \frac{1}{2}\log_2\left(\frac{1}{2}\right)$$

$$= \frac{1}{2} + \frac{1}{2} = 1$$

Quantifying disorder

How does D looks like as function of P/T?

$$D(set) = -\frac{P}{T}\log_2\left(\frac{P}{T}\right) - \frac{N}{T}\log_2\left(\frac{N}{T}\right)$$



If P=N?

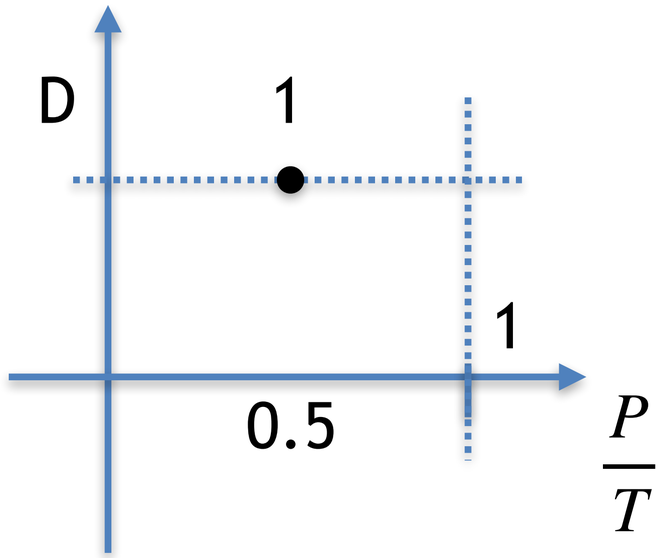
$$D(set) = -\frac{1}{2}\log_2\left(\frac{1}{2}\right) - \frac{1}{2}\log_2\left(\frac{1}{2}\right)$$

$$= \frac{1}{2} + \frac{1}{2} = 1$$

Quantifying disorder

How does D looks like as function of P/T?

$$D(set) = -\frac{P}{T}\log_2\left(\frac{P}{T}\right) - \frac{N}{T}\log_2\left(\frac{N}{T}\right)$$



If P=0? $D(set) = 0$

If P=T? $D(set) = 0$

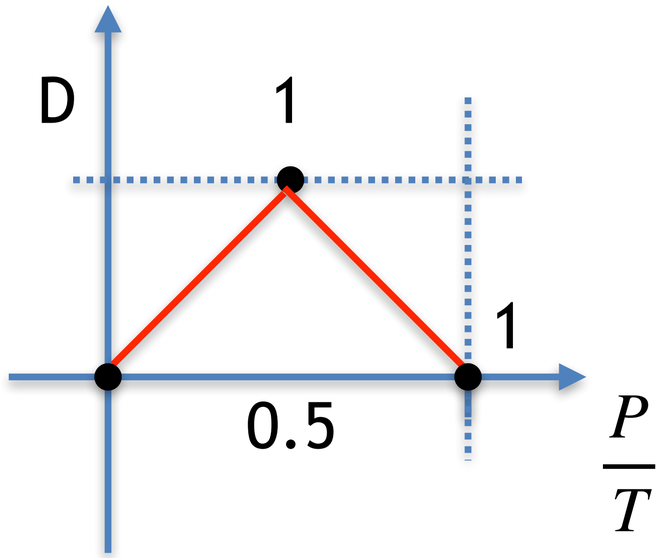
Quantifying disorder

How does D looks like as function of P/T?

$$D(set) = -\frac{P}{T}\log_2\left(\frac{P}{T}\right) - \frac{N}{T}\log_2\left(\frac{N}{T}\right)$$

If P=0? $D(set) = 0$

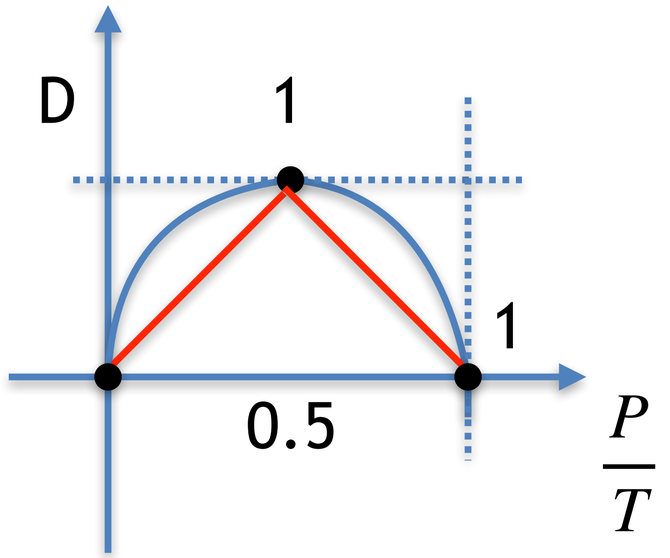
If P=T? $D(set) = 0$



Quantifying disorder

How does D looks like as function of P/T?

$$D(set) = -\frac{P}{T}\log_2\left(\frac{P}{T}\right) - \frac{N}{T}\log_2\left(\frac{N}{T}\right)$$



If P=0? $D(set) = 0$

If P=T? $D(set) = 0$

Quantifying disorder

So the quality of a test can we written as

$$Q = \sum_{s} D(s)$$

Quantifying disorder

So the quality of a test can we written as

$$Q = \sum_{s} D(s)$$

It is however better to have a normalized quantity according to the number of samples

Quantifying disorder

So the quality of a test can we written as

$$Q = \sum_s D(s)$$

It is however better to have a normalized quantity according to the number of samples

$$Q = \sum_s D(s)\frac{N_s}{N_t}$$

Quantifying disorder

So the quality of a test can we written as

$$Q = \sum_s D(s)$$

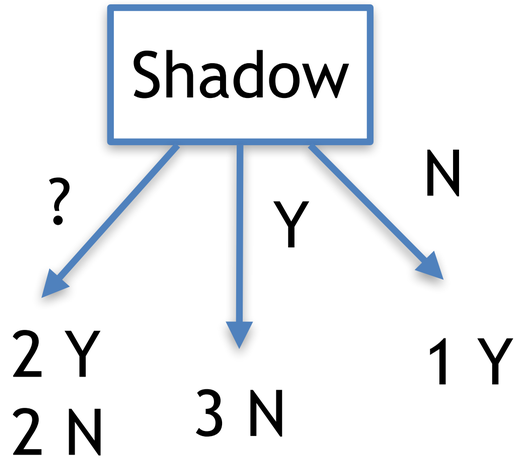
It is however better to have a normalized quantity according to the number of samples

$$Q = \sum_s D(s)\frac{N_s}{N_t}$$

Where we have N_s as the number of samples in the set and N_t as the total number of samples that enters the test

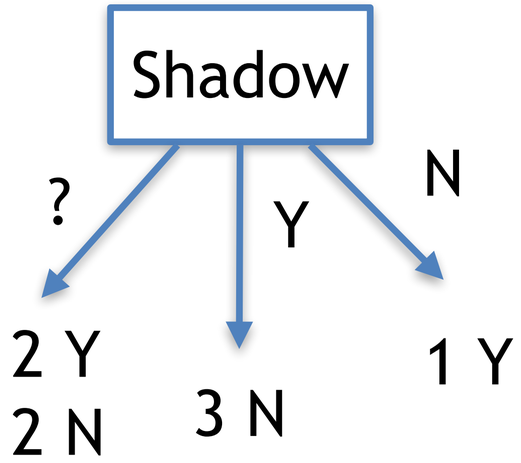
Identification trees

What is the value of Q in this case? In this case, the smallest the better!



Identification trees

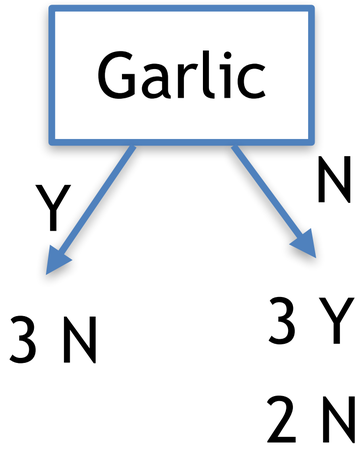
What is the value of Q in this case? In this case, the smallest the better!



$$Q = \sum_s D(s)\frac{N_s}{N_t} = 1 \times \frac{4}{8} + 0 \times \frac{3}{8} + 0 \times \frac{1}{8} = \frac{1}{2}$$

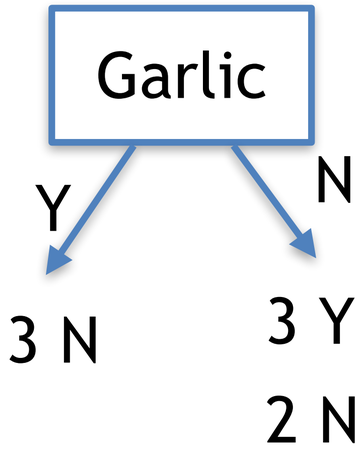
Identification trees

What is the value of Q in this case?



Identification trees

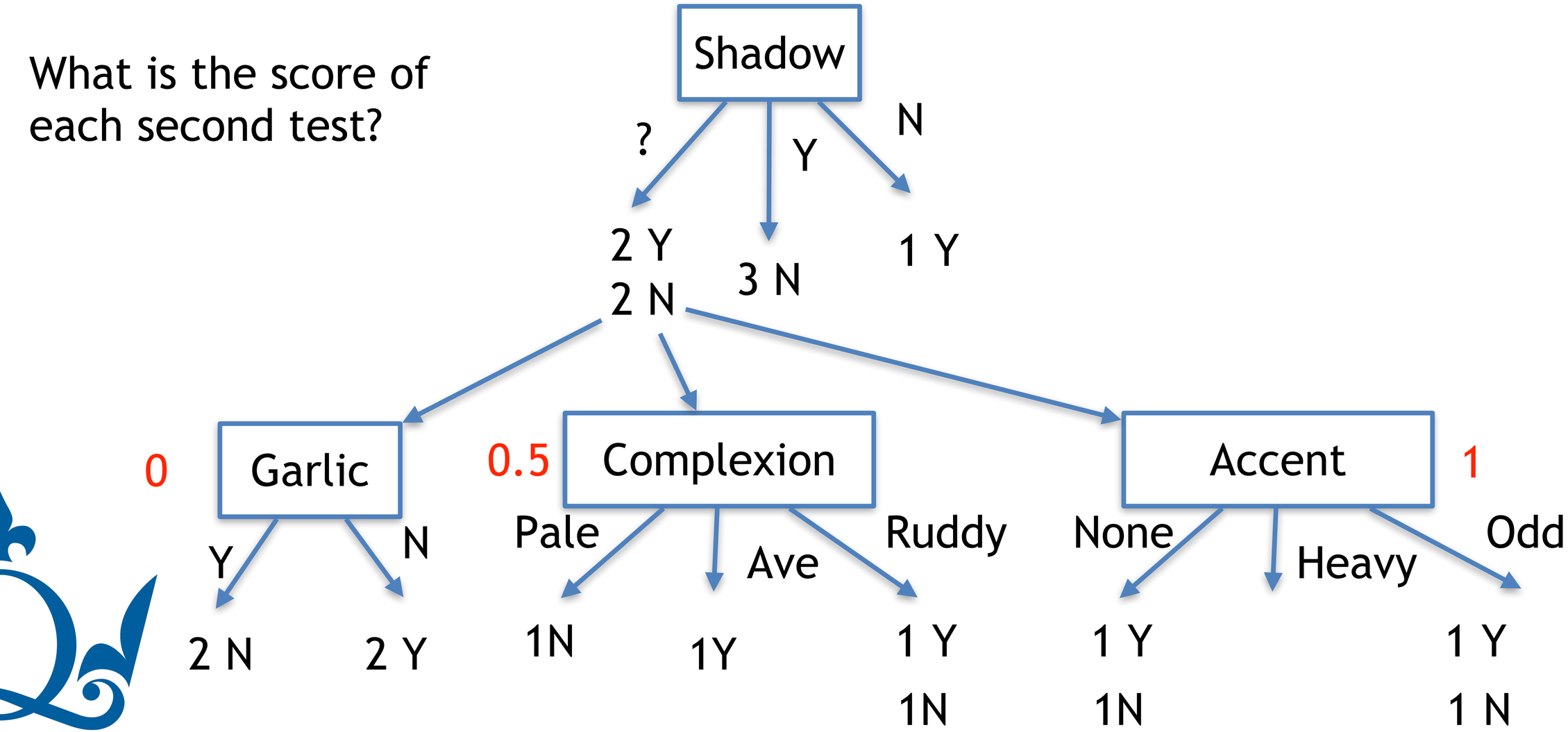
What is the value of Q in this case?



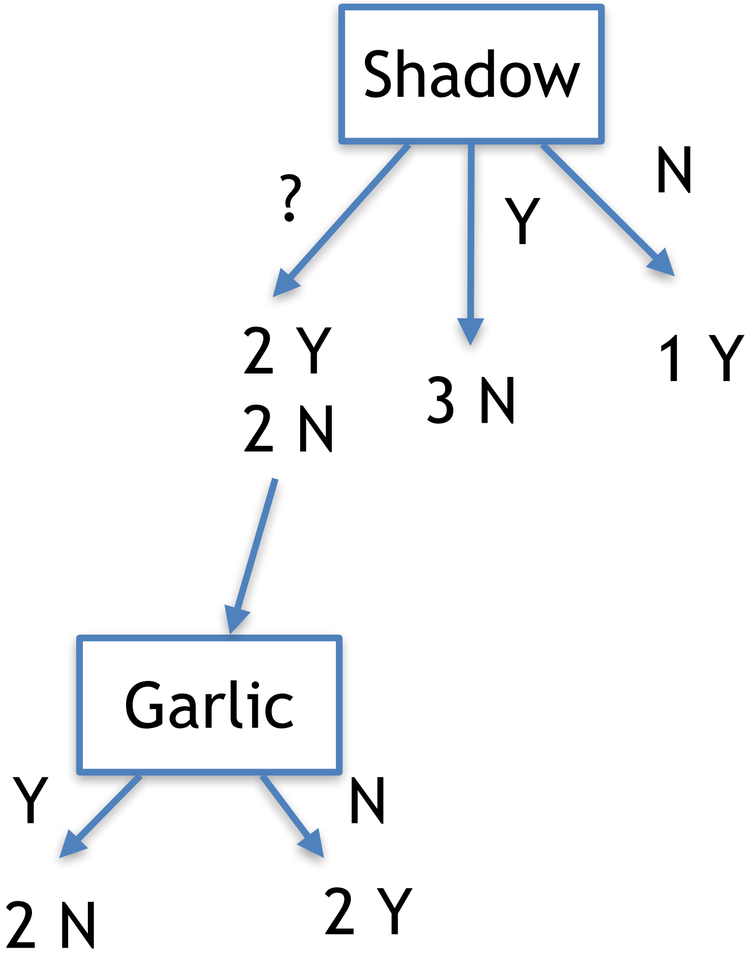
$$Q = \sum_s D(s)\frac{N_s}{N_t} = 0 \times \frac{3}{8} + 0.97 \times \frac{5}{8} \sim 0.61$$

Identification trees

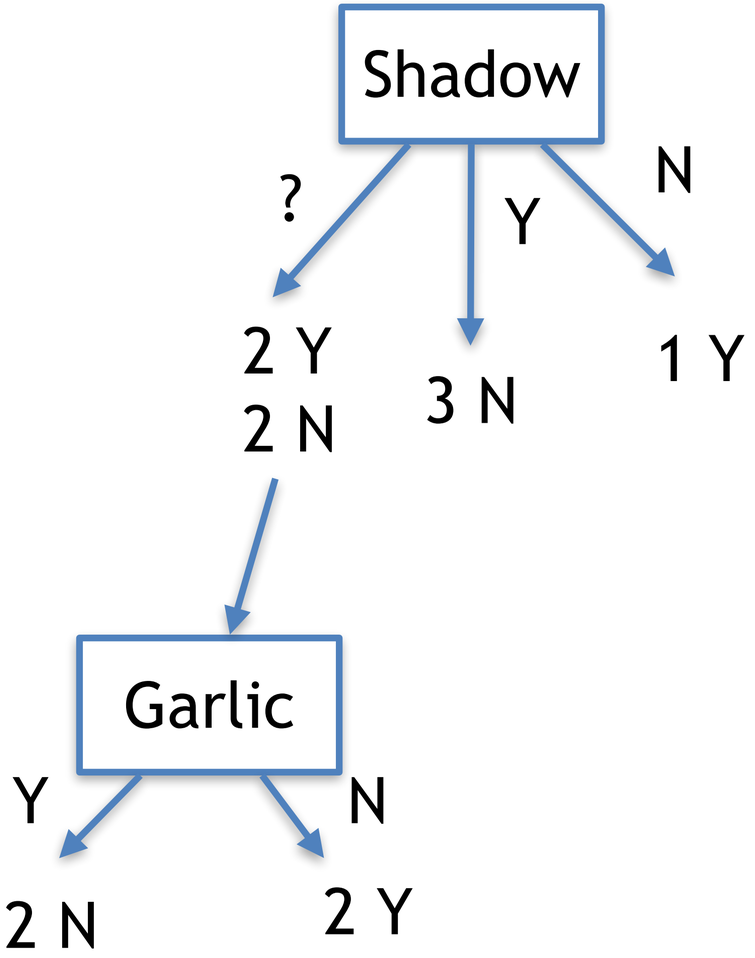
What is the score of each second test?



Identification trees

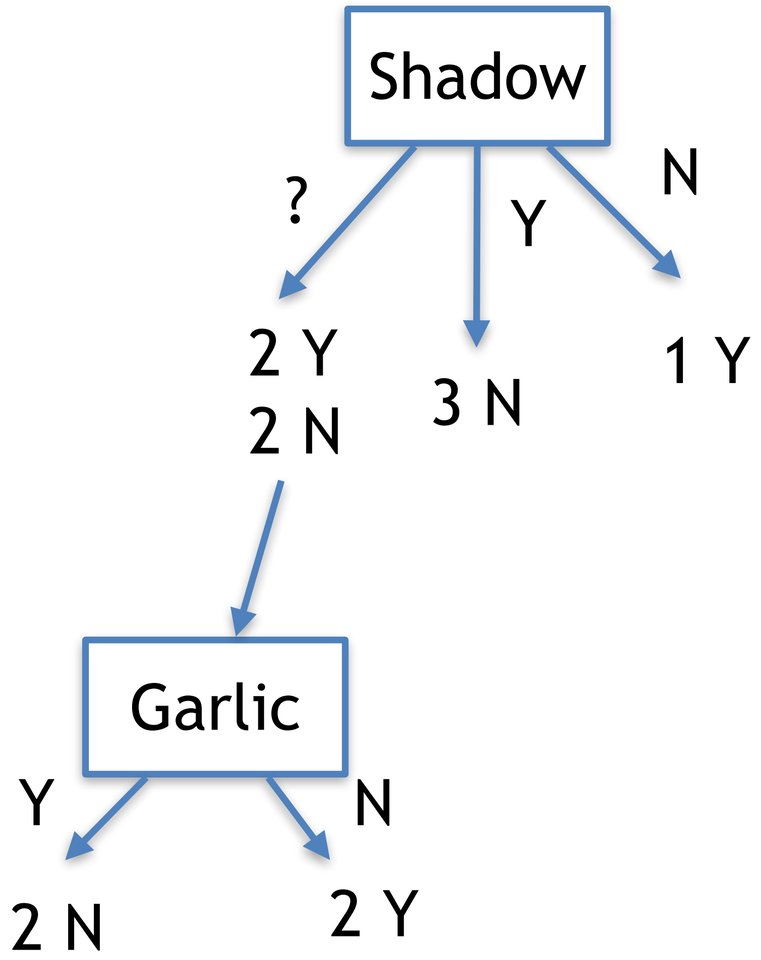


Identification trees



We can convert this tree into a set of rules to classify

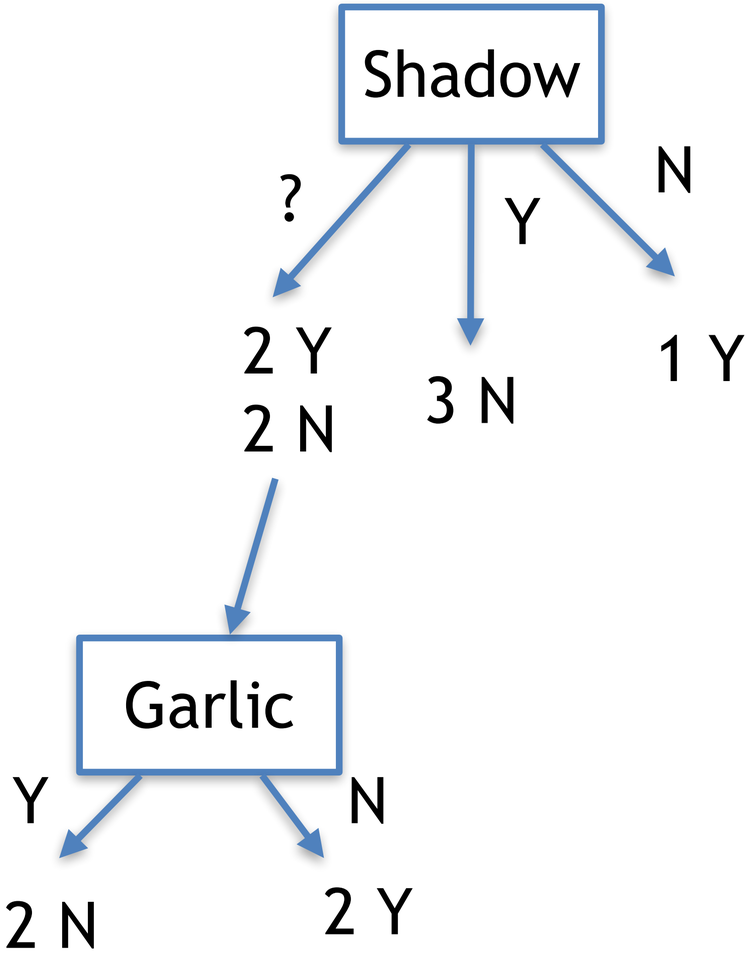
Identification trees



We can convert this tree into a set of rules to classify

If Shadow ? and Garlic Y -> not a vampire

Identification trees

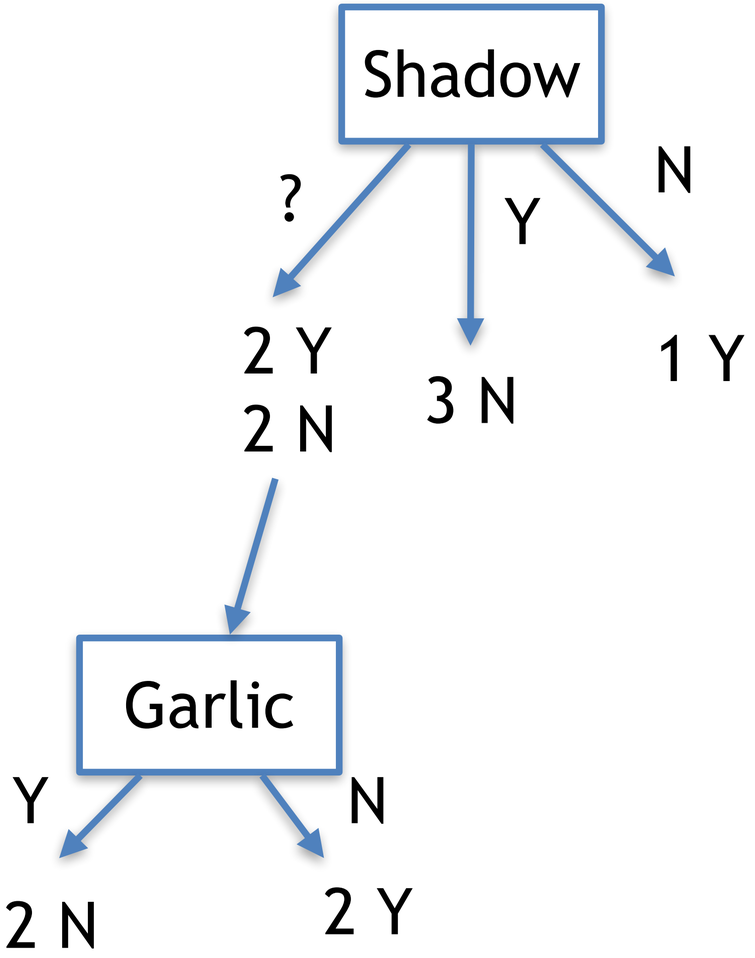


We can convert this tree into a set of rules to classify

If Shadow ? and Garlic Y -> not a vampire

If Shadow ? and Garlic N -> vampire

Identification trees



We can convert this tree into a set of rules to classify

If Shadow ? and Garlic Y -> not a vampire

If Shadow ? and Garlic N -> vampire

Note: we might not need both rules in practice
In our data if Garlic is N we have a vampire independently of the Shadow

Quantifying disorder

There are also other metrics such as the **Gini impurity**

Quantifying disorder

There are also other metrics such as the **Gini impurity**

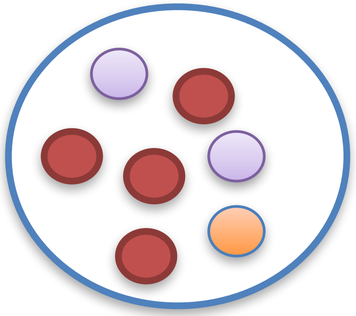
Let us consider a group of N items

Quantifying disorder

There are also other metrics such as the **Gini impurity**

Let us consider a group of N items

Let us assume that they belong to one of K categories

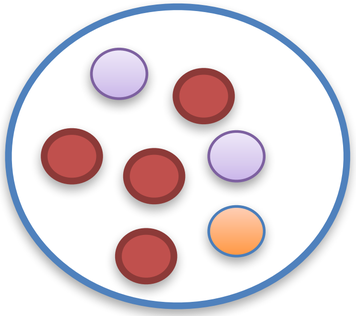


Quantifying disorder

There are also other metrics such as the **Gini impurity**

Let us consider a group of N items

Let us assume that they belong to one of K categories

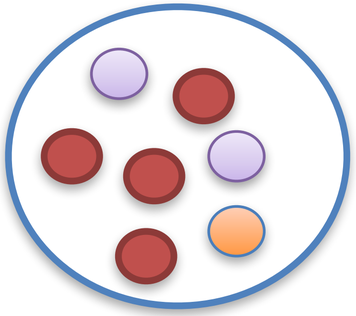


We can define the probability p_i for each category as $p_i = \frac{N_i}{N}$

Quantifying disorder

There are also other metrics such as the **Gini impurity**

Let us consider a group of N items



Let us assume that they belong to one of K categories

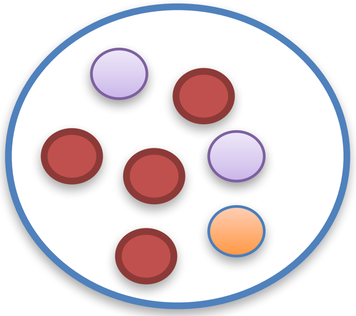
We can define the probability p_i for each category as $p_i = \frac{N_i}{N}$

The Gini impurity of this population is then defined as

Quantifying disorder

There are also other metrics such as the **Gini impurity**

Let us consider a group of N items



Let us assume that they belong to one of K categories

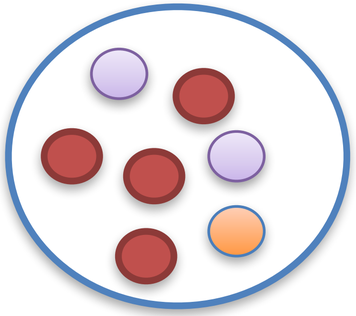
We can define the probability p_i for each category as $p_i = \frac{N_i}{N}$

The Gini impurity of this population is then defined as

$$GI = 1 - \sum_{i=1}^{k} p_i^2$$

Quantifying disorder

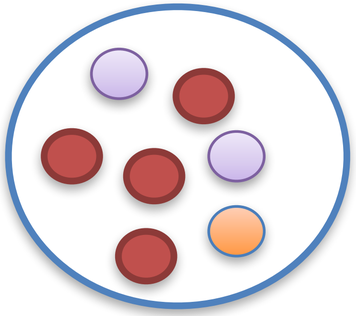
In this example $N = 7$



Quantifying disorder

In this example $N = 7$

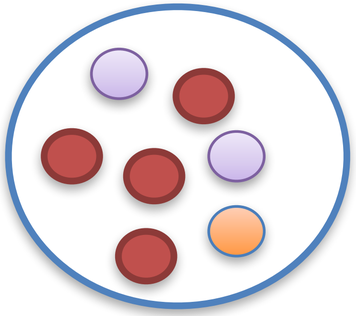
$$p_{red} = \frac{4}{7}$$



Quantifying disorder

In this example $N = 7$

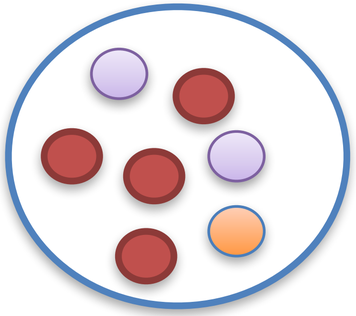
$$p_{red} = \frac{4}{7} \qquad p_{violet} = \frac{2}{7}$$



Quantifying disorder

In this example $N = 7$

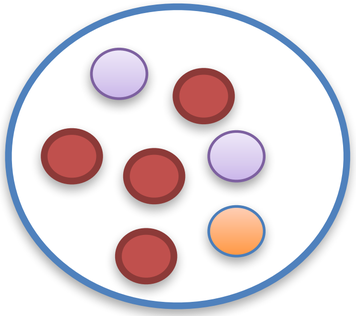
$$p_{red} = \frac{4}{7} \qquad p_{violet} = \frac{2}{7} \qquad p_{orange} = \frac{1}{7}$$



Quantifying disorder

In this example $N = 7$

$$p_{red} = \frac{4}{7} \qquad p_{violet} = \frac{2}{7} \qquad p_{orange} = \frac{1}{7}$$

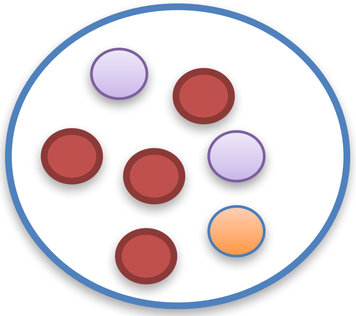


Thus

Quantifying disorder

In this example $N = 7$

$$p_{red} = \frac{4}{7} \qquad p_{violet} = \frac{2}{7} \qquad p_{orange} = \frac{1}{7}$$



Thus

$$GI = 1 - \left(\frac{4}{7}\right)^2 - \left(\frac{2}{7}\right)^2 - \left(\frac{1}{7}\right)^2 = 1 - \frac{21}{49} = 0.57$$

Quantifying disorder

In this example, they are all different $K = N = 7$

Quantifying disorder

In this example, they are all different $K = N = 7$

$$p_i = \frac{1}{N}$$

Quantifying disorder

In this example, they are all different $K = N = 7$

$$p_i = \frac{1}{N}$$

Thus

Quantifying disorder

In this example, they are all different $K = N = 7$

$$p_i = \frac{1}{N}$$

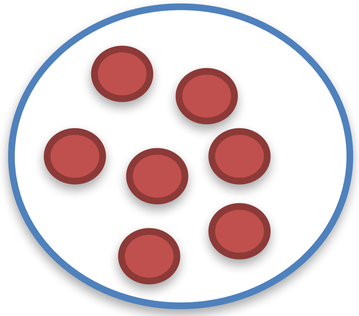
Thus

$$GI = 1 - \sum_{i=1}^{K} \frac{1}{N^2} = 1 - \frac{K}{N^2} = 1 - \frac{N}{N^2} = 1 - \frac{1}{N} = \frac{N-1}{N} \sim 1$$

Quantifying disorder

In this example, they are all the same

$N = 7$ $K = 1$

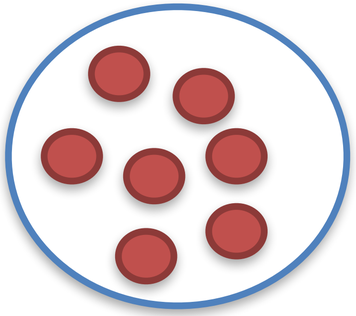


Quantifying disorder

In this example, they are all the same

$N = 7$ $K = 1$

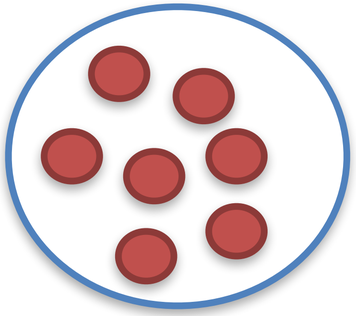
$$p_1 = 1$$



Quantifying disorder

In this example, they are all the same

$N = 7$ $K = 1$



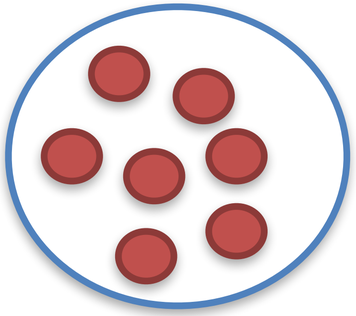
$$p_1 = 1$$

Thus

Quantifying disorder

In this example, they are all the same

$N = 7$ $K = 1$



$$p_1 = 1$$

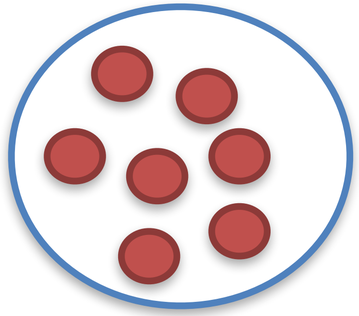
Thus

$$GI = 1 - p_1^2 = 0$$

Quantifying disorder

In this example, they are all the same

$N = 7$ $K = 1$



$$p_1 = 1$$

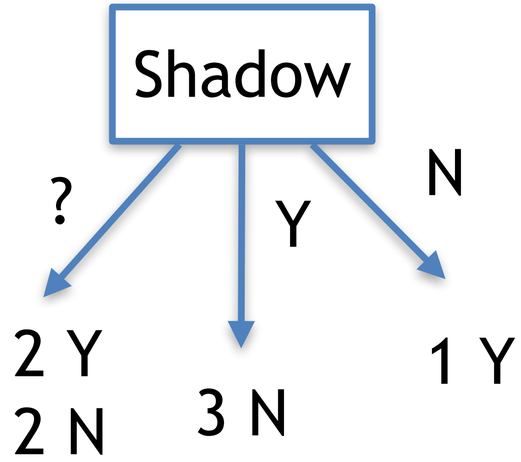
Thus

$$GI = 1 - p_1^2 = 0$$

Hence, the Gini impurity is zero for “pure” samples and 1 for maximally “impure”

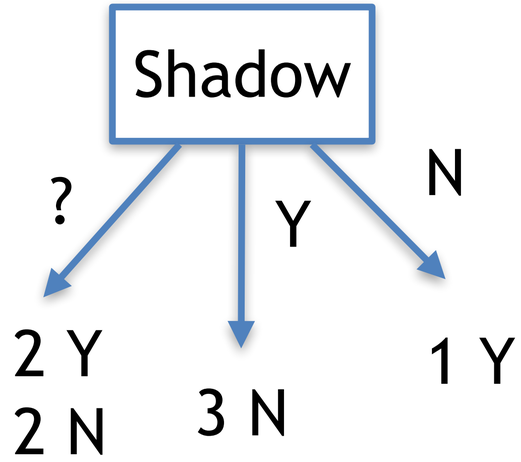
Quantifying disorder

What is the Gini impurity of the outcome of each test?



Quantifying disorder

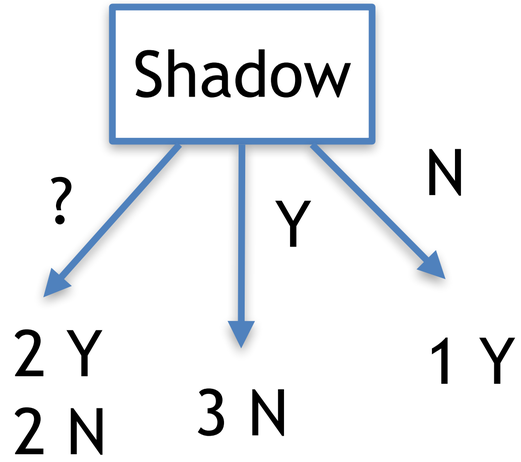
What is the Gini impurity of the outcome of each test?



We have, in this case, three groups (leafs), we start by computing their GI

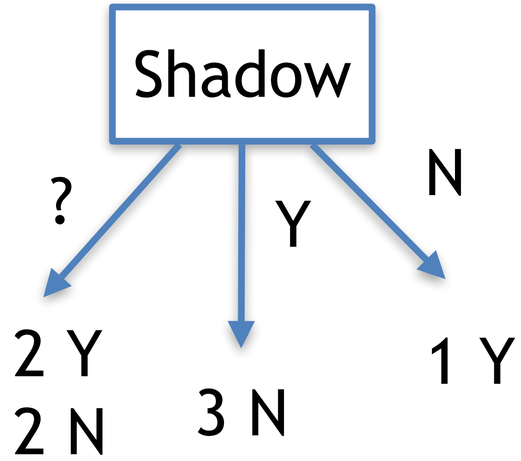
Quantifying disorder

What is the Gini impurity of the outcome of each test?



Quantifying disorder

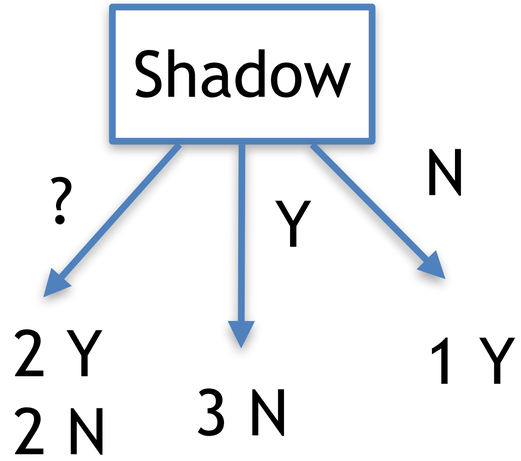
What is the Gini impurity of the outcome of each test?



$$GI_{?} = 1 - \left(\frac{2}{4}\right)^2 - \left(\frac{2}{4}\right)^2 = \frac{1}{2}$$

Quantifying disorder

What is the Gini impurity of the outcome of each test?

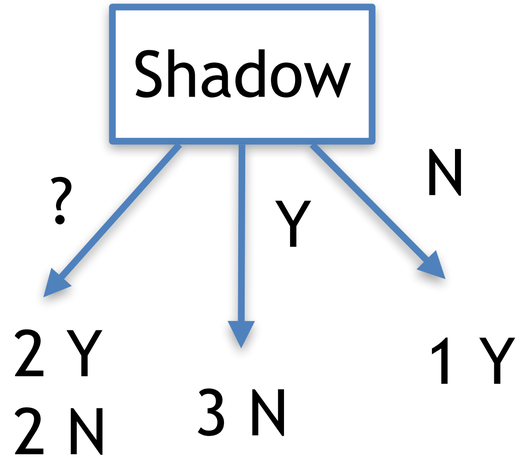


$$GI_? = 1 - \left(\frac{2}{4}\right)^2 - \left(\frac{2}{4}\right)^2 = \frac{1}{2}$$

$$GI_Y = 0$$

Quantifying disorder

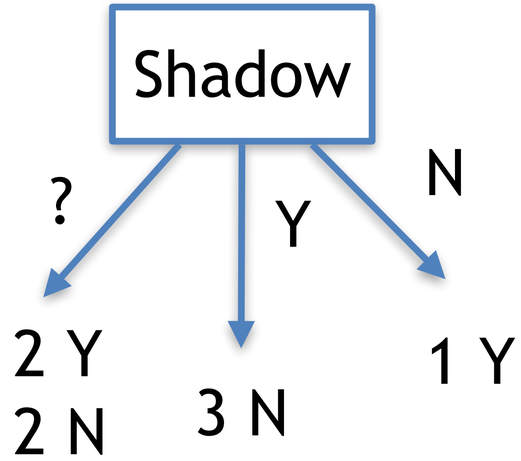
What is the Gini impurity of the outcome of each test?



$$GI_? = 1 - \left(\frac{2}{4}\right)^2 - \left(\frac{2}{4}\right)^2 = \frac{1}{2} \qquad GI_Y = 0 \qquad GI_N = 0$$

Quantifying disorder

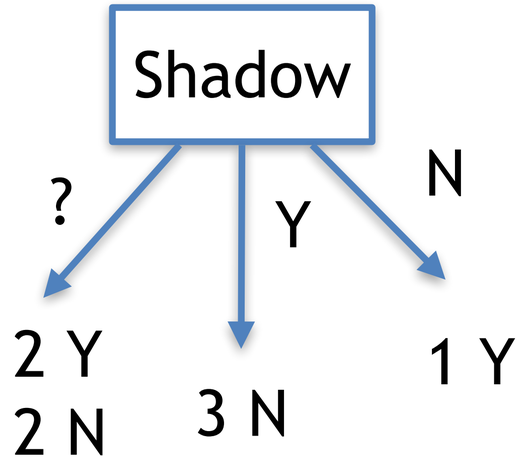
What is the Gini impurity of the outcome of each test?



$$GI_{?} = 1 - \left(\frac{2}{4}\right)^2 - \left(\frac{2}{4}\right)^2 = \frac{1}{2} \qquad GI_Y = 0 \qquad GI_N = 0$$

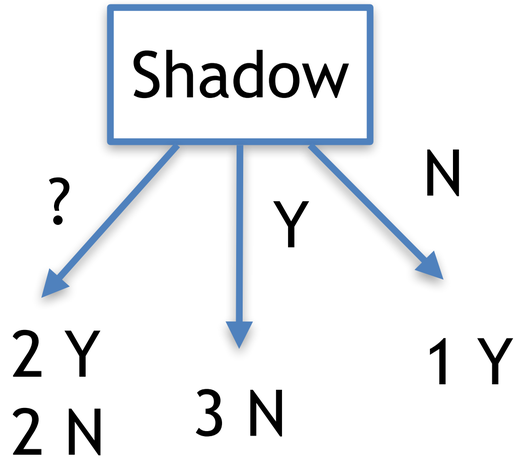
We can combined them using a weighted average, considering the different number of people in each leaf

Quantifying disorder



$$GI_{shadow} = w_? GI_? + w_Y GI_Y + w_N GI_N$$

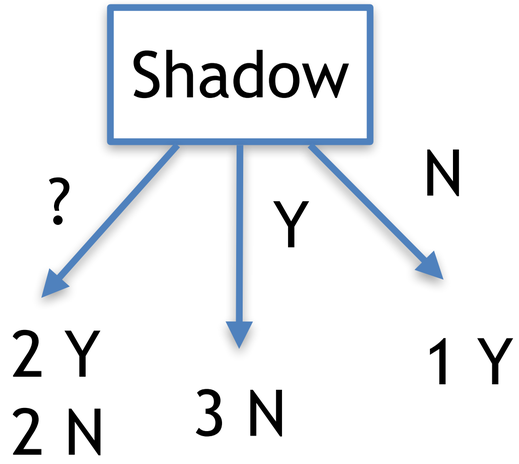
Quantifying disorder



$$GI_{shadow} = w_? GI_? + w_Y GI_Y + w_N GI_N$$

$$w_? = \frac{4}{8} = \frac{1}{2}$$

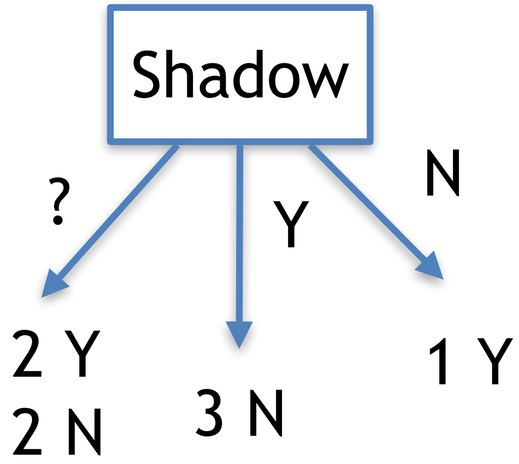
Quantifying disorder



$$GI_{shadow} = w_? GI_? + w_Y GI_Y + w_N GI_N$$

$$w_? = \frac{4}{8} = \frac{1}{2} \qquad w_Y = \frac{3}{8}$$

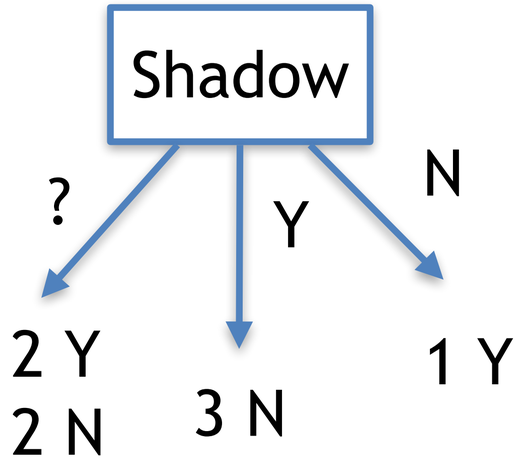
Quantifying disorder



$$GI_{shadow} = w_? GI_? + w_Y GI_Y + w_N GI_N$$

$$w_? = \frac{4}{8} = \frac{1}{2} \qquad w_Y = \frac{3}{8} \qquad w_N = \frac{1}{8}$$

Quantifying disorder



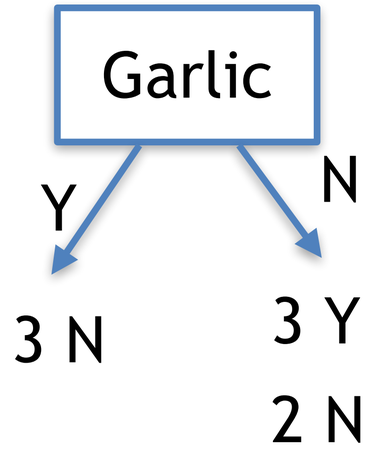
$$GI_{shadow} = w_? GI_? + w_Y GI_Y + w_N GI_N$$

$$w_? = \frac{4}{8} = \frac{1}{2} \qquad w_Y = \frac{3}{8} \qquad w_N = \frac{1}{8}$$

$$GI_{shadow} = w_? GI_? = \frac{1}{4}$$

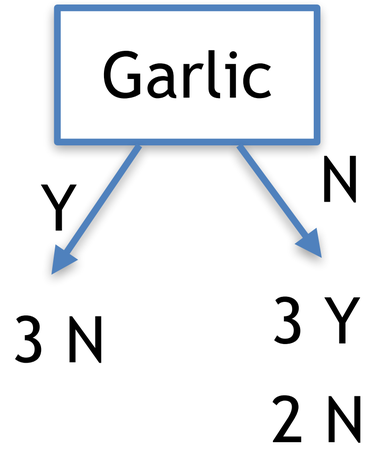
Quantifying disorder

What is the Gini impurity of the outcome of each test?



Quantifying disorder

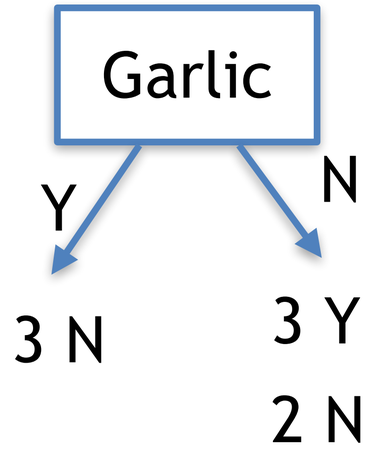
What is the Gini impurity of the outcome of each test?



$$GI_Y = 0$$

Quantifying disorder

What is the Gini impurity of the outcome of each test?

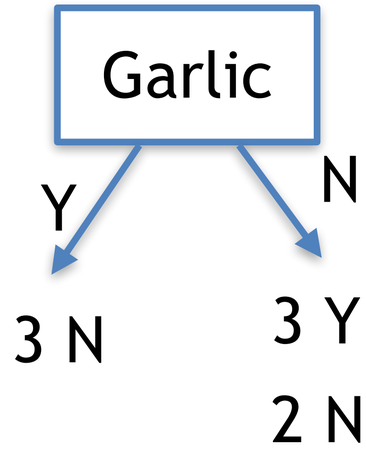


$$GI_Y = 0$$

$$GI_N = 1 - \left(\frac{3}{5}\right)^2 - \left(\frac{2}{5}\right)^2 = 0.48$$

Quantifying disorder

What is the Gini impurity of the outcome of each test?



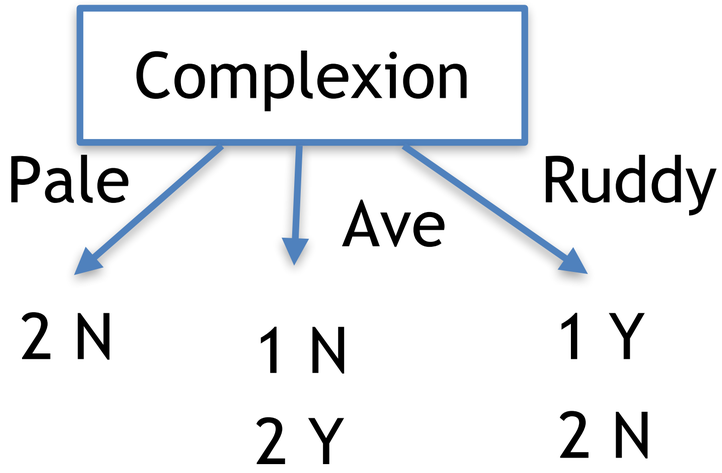
$$GI_Y = 0$$

$$GI_N = 1 - \left(\frac{3}{5}\right)^2 - \left(\frac{2}{5}\right)^2 = 0.48$$

$$GI_{garlic} = w_N GI_N + w_Y GI_Y = \frac{5}{8} 0.48 = 0.3$$

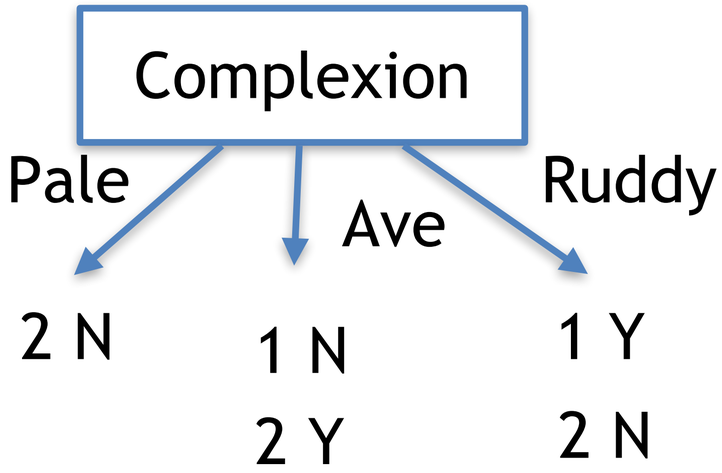
Quantifying disorder

What is the Gini impurity of the outcome of each test?



Quantifying disorder

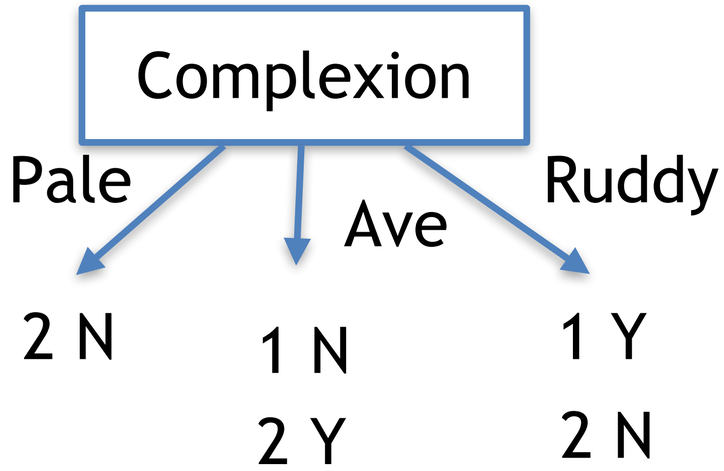
What is the Gini impurity of the outcome of each test?



$$GI_{Pale} = 0$$

Quantifying disorder

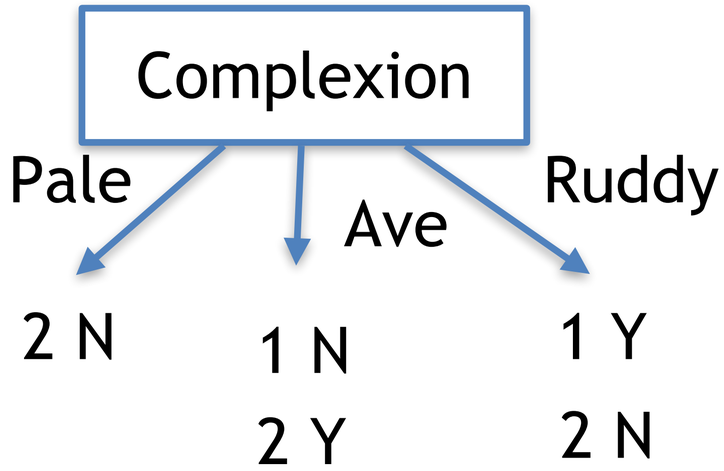
What is the Gini impurity of the outcome of each test?



$$GI_{Pale} = 0 \qquad GI_{ave} = 1 - \left(\frac{1}{3}\right)^2 - \left(\frac{2}{3}\right)^2 = 0.44$$

Quantifying disorder

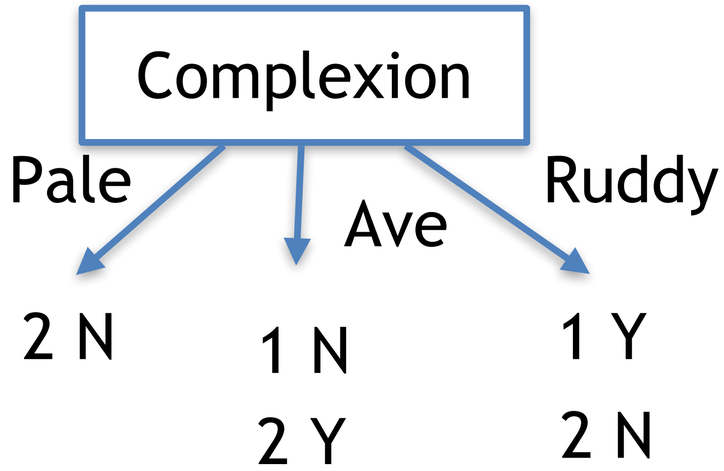
What is the Gini impurity of the outcome of each test?



$$GI_{Pale} = 0 \qquad GI_{ave} = 1 - \left(\frac{1}{3}\right)^2 - \left(\frac{2}{3}\right)^2 = 0.44 \qquad GI_{ruddy} = 0.44$$

Quantifying disorder

What is the Gini impurity of the outcome of each test?

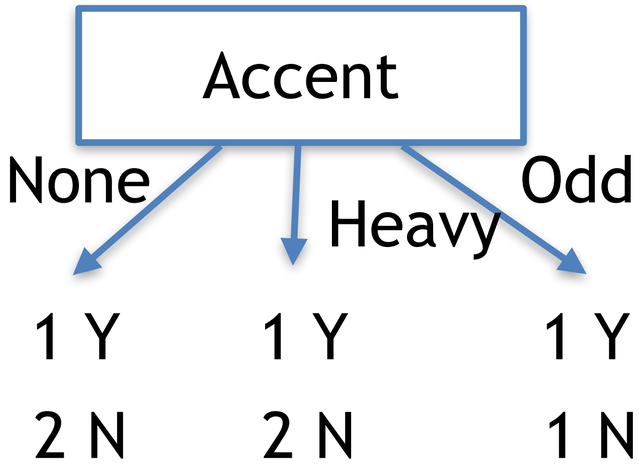


$$GI_{Pale} = 0 \qquad GI_{ave} = 1 - \left(\frac{1}{3}\right)^2 - \left(\frac{2}{3}\right)^2 = 0.44 \qquad GI_{ruddy} = 0.44$$

$$GI_{complexion} = 2\frac{3}{8}0.44 = 0.33$$

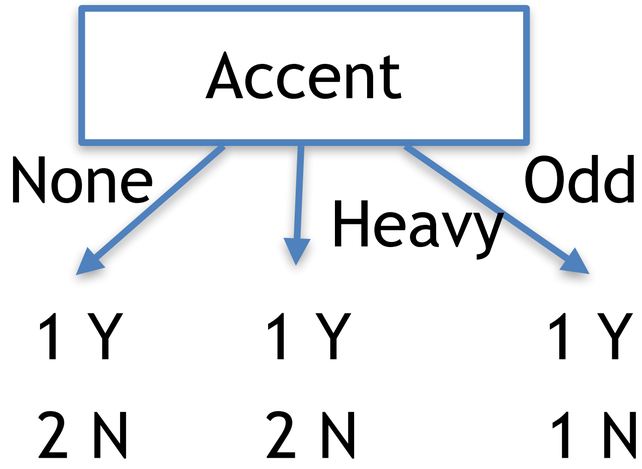
Quantifying disorder

What is the Gini impurity of the outcome of each test?



Quantifying disorder

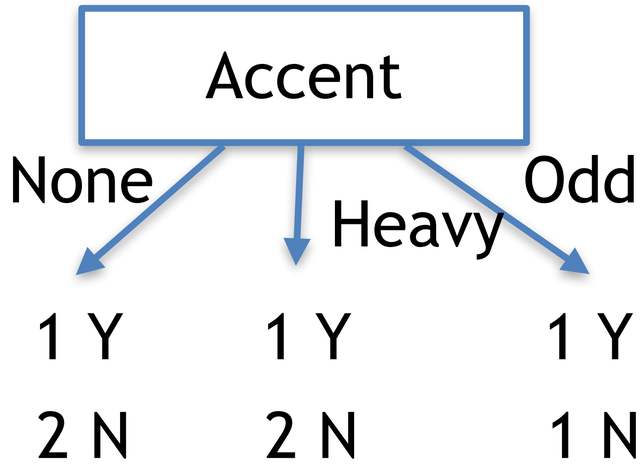
What is the Gini impurity of the outcome of each test?



$$GI_{none} = 0.44$$

Quantifying disorder

What is the Gini impurity of the outcome of each test?

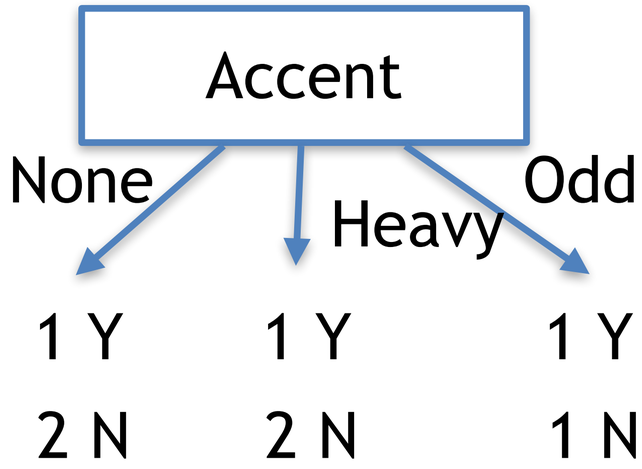


$$GI_{none} = 0.44$$

$$GI_{heavy} = 0.44$$

Quantifying disorder

What is the Gini impurity of the outcome of each test?



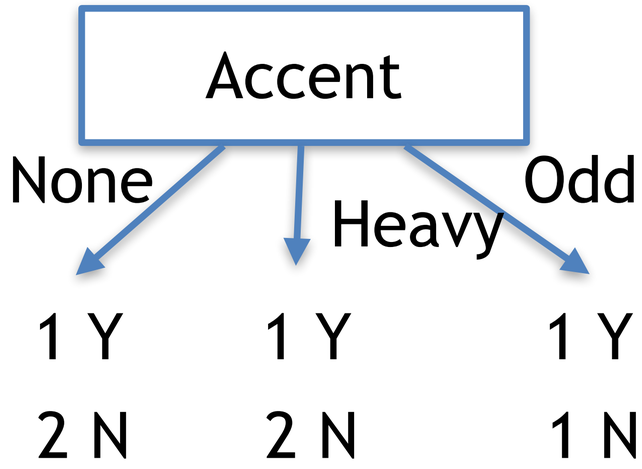
$$GI_{none} = 0.44$$

$$GI_{heavy} = 0.44$$

$$GI_{odd} = 0.5$$

Quantifying disorder

What is the Gini impurity of the outcome of each test?

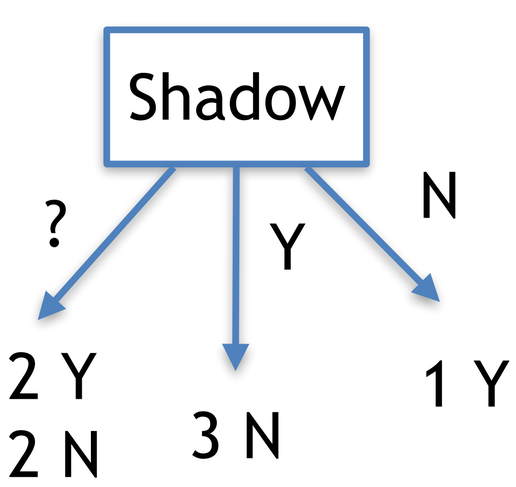


$$GI_{none} = 0.44 \qquad GI_{heavy} = 0.44 \qquad GI_{odd} = 0.5$$

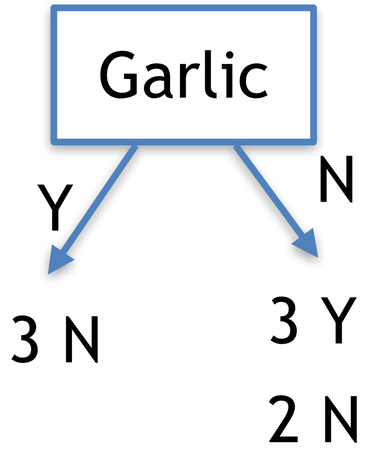
$$GI_{complexion} = 2\frac{3}{8}0.44 + \frac{2}{8}0.5 = 0.45$$

Identification trees

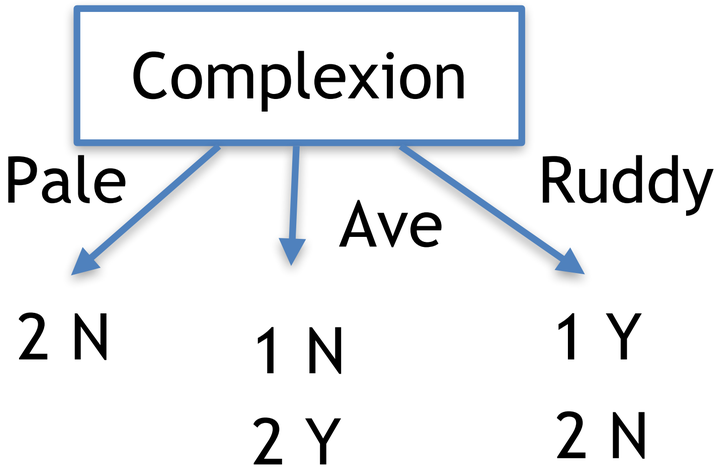
Shadow appears to be the best also according to this score



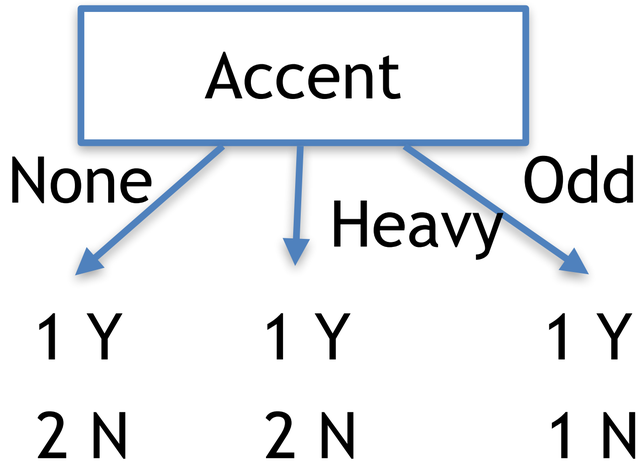
0.25



0.3



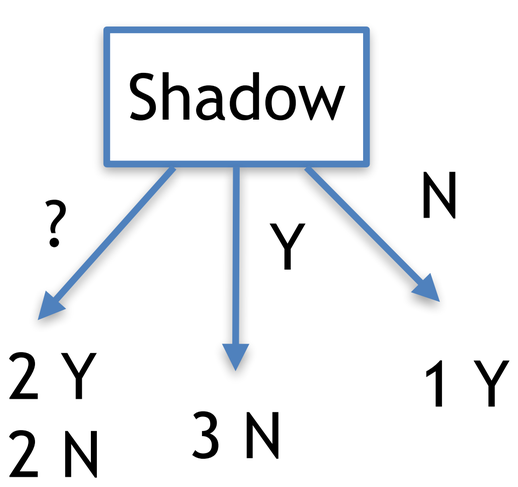
0.33



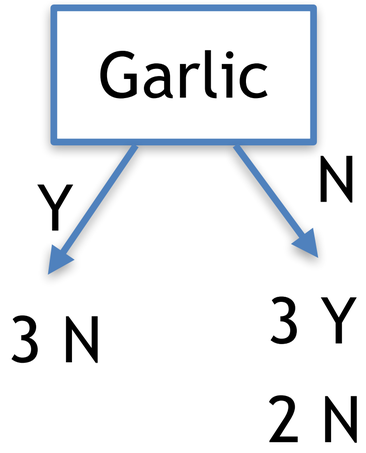
0.45

Identification trees

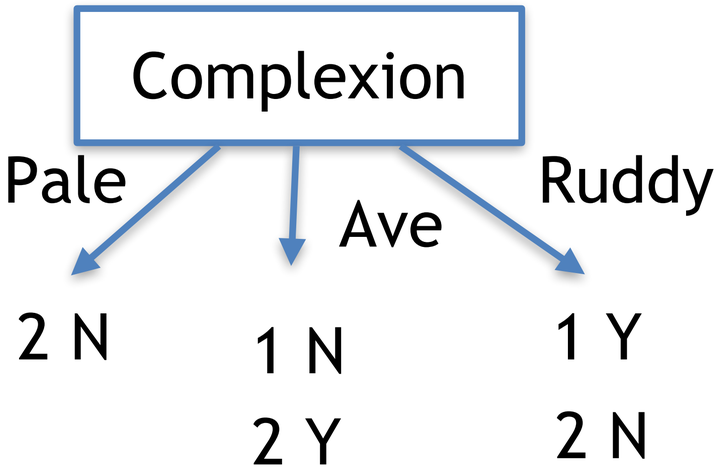
Shadow appears to be the best also according to this score



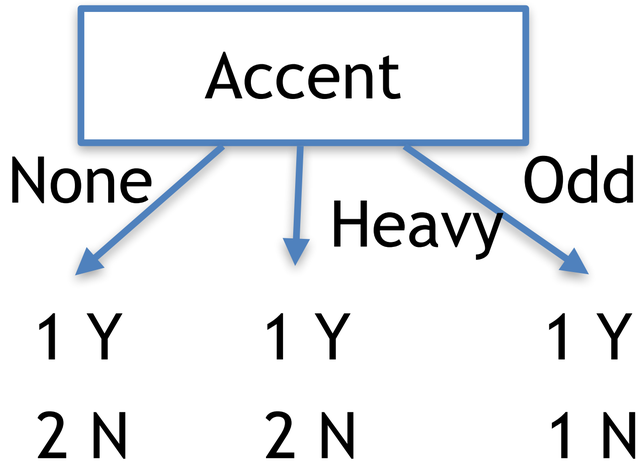
0.25



0.3



0.33

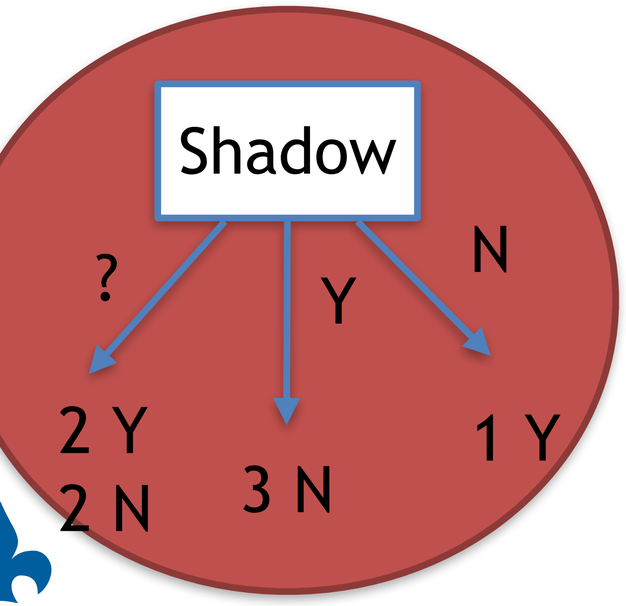


0.45

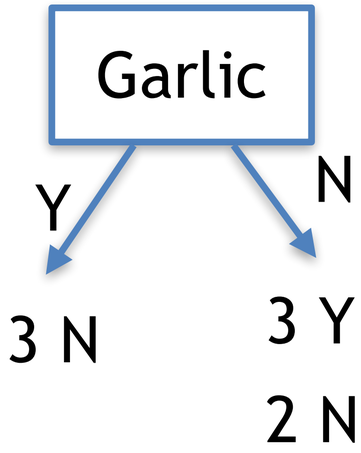
We can then select it as the root of the tree

Identification trees

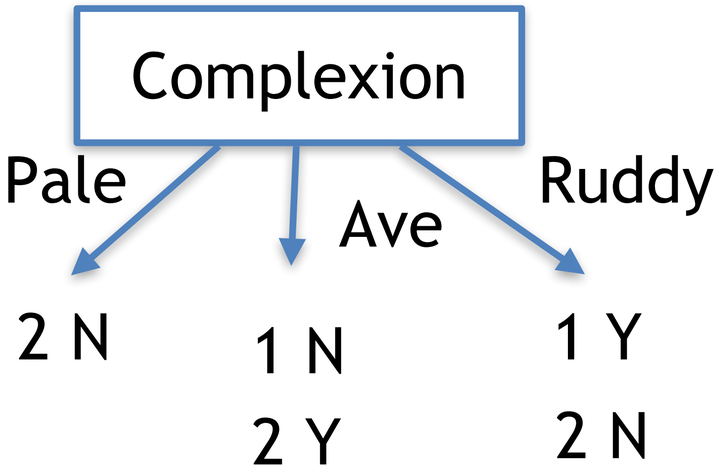
Shadow appears to be the best also according to this score



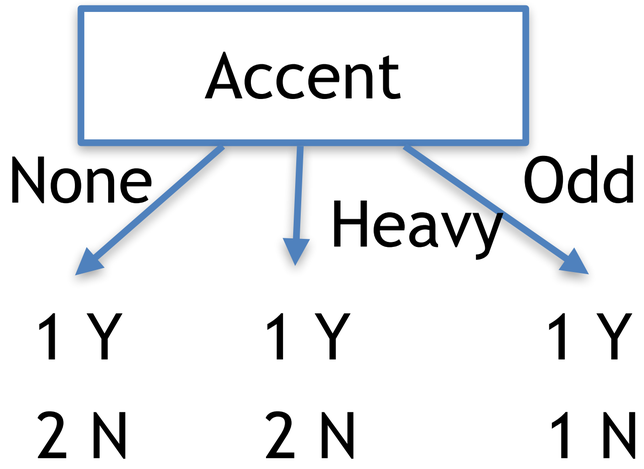
0.25



0.3



0.33



0.45

We can then select it as the root of the tree

Identification trees

What if we have numeric values? Can we use the method?

Identification trees

What if we have numeric values? Can we use the method?

Imagine that we have another variable that takes numeric values

Identification trees

What if we have numeric values? Can we use the method?

Imagine that we have another variable that takes numeric values

Identification trees

What if we have numeric values? Can we use the method?

Imagine that we have another variable that takes numeric values

We can set a threshold and transform the variable in categorical

Identification trees

What if we have numeric values? Can we use the method?

Imagine that we have another variable that takes numeric values

We can set a threshold and transform the variable in categorical

How do I select such threshold?

Identification trees

Imagine that we get another variable

Shadow	Garlic	Complexion	Accent	Height	Vampire
?	Yes	Pale	None	162	No
Yes	Yes	Ruddy	None	173	No
?	No	Ruddy	None	198	Yes
No	No	Average	Heavy	185	Yes
?	No	Average	Odd	197	Yes
Yes	No	Pale	Heavy	187	No
Yes	No	Average	Heavy	175	No
?	Yes	Ruddy	Odd	168	No

Identification trees

We first order the data according to the numeric values

Height	Vampire
162	No
168	No
173	No
175	No
185	Yes
187	Yes
197	Yes
198	Yes

Identification trees

Then we compute the average for each pair

Height	Vampire
162	No
168	No
173	No
175	No
185	Yes
187	Yes
197	Yes
198	Yes

Identification trees

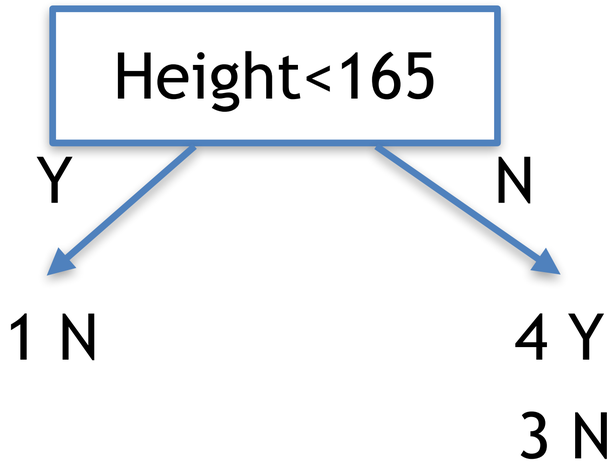
Then we compute the average for each pair

	Height	Vampire
	162	No
165		
	168	No
170.5		
	173	No
174		
	175	No
180		
	185	Yes
186		
	187	Yes
192		
	197	Yes
197.5		
	198	Yes

Identification trees

The we use each average as threshold and compute the disorder of the test done with that value

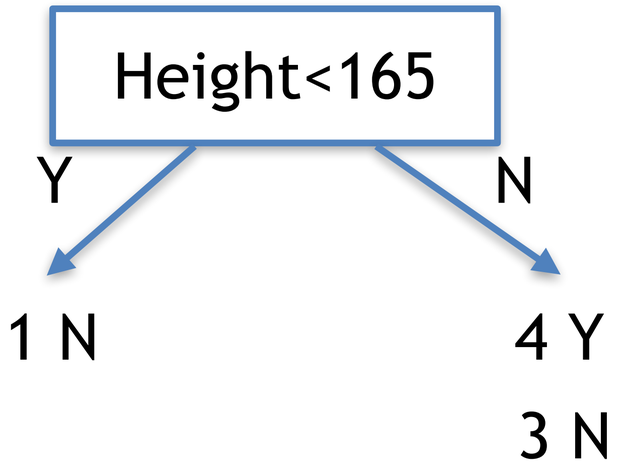
	Height	Vampire
165	162	No
170.5	168	No
	173	No
174	175	No
180	185	Yes
186	187	Yes
192	197	Yes
197.5	198	Yes



Identification trees

The we use each average as threshold and compute the disorder of the test done with that value

	Height	Vampire
165	162	No
170.5	168	No
	173	No
174	175	No
180	185	Yes
186	187	Yes
192	197	Yes
197.5	198	Yes

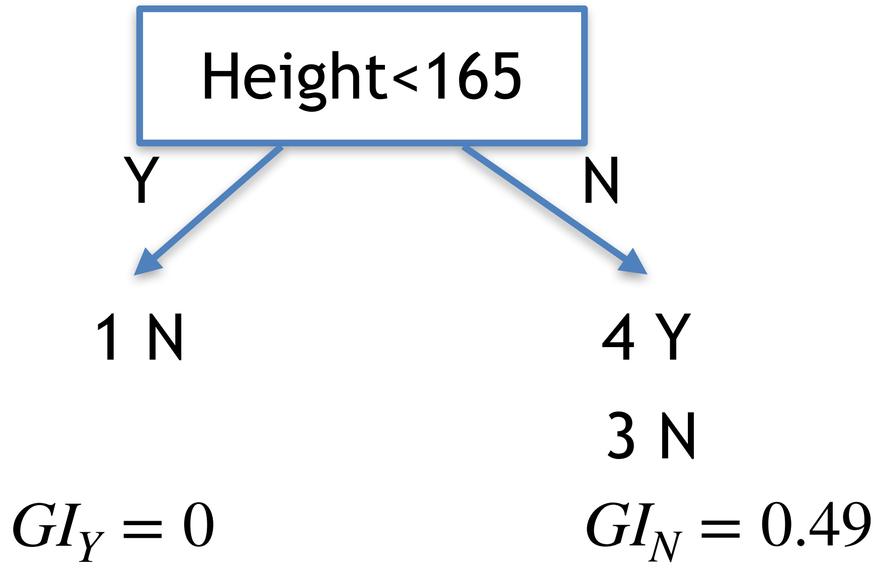


$$GI_Y = 0$$

Identification trees

The we use each average as threshold and compute the disorder of the test done with that value

	Height	Vampire
165	162	No
170.5	168	No
	173	No
174	175	No
180	185	Yes
186	187	Yes
192	197	Yes
197.5	198	Yes



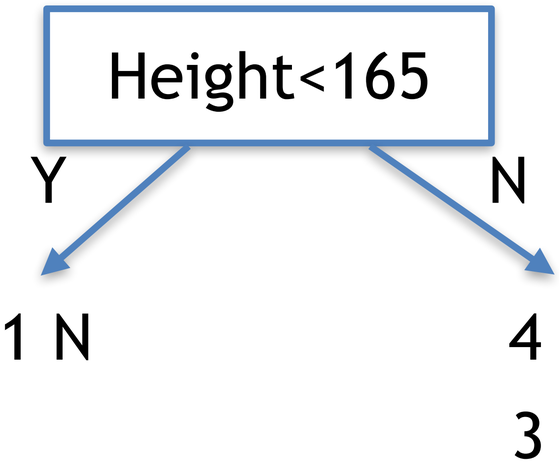
$GI_Y = 0$

$GI_N = 0.49$

Identification trees

The we use each average as threshold and compute the disorder of the test done with that value

	Height	Vampire
165	162	No
170.5	168	No
	173	No
174	175	No
180	185	Yes
186	187	Yes
192	197	Yes
197.5	198	Yes



$$GI_Y = 0 \qquad GI_N = 0.49$$

$$GI_{Height<165} = \frac{7}{8}0.49 = 0.42$$

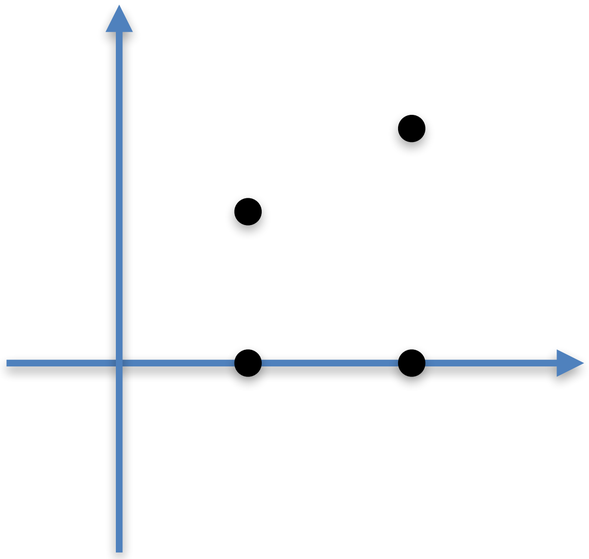
Identification trees

We then try this with all possible values and get the smallest!

	Height	Vampire
165	162	No
170.5	168	No
	173	No
174	175	No
180	185	Yes
186	187	Yes
192	197	Yes
197.5	198	Yes

Identification trees

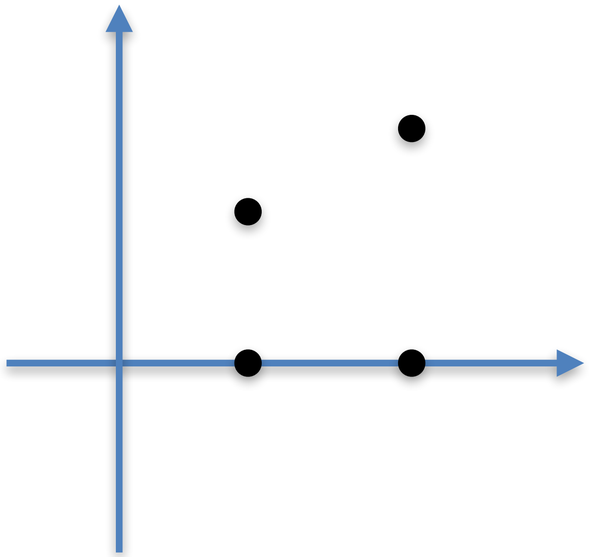
Often numeric data is represented in this way



Identification trees

Often numeric data is represented in this way

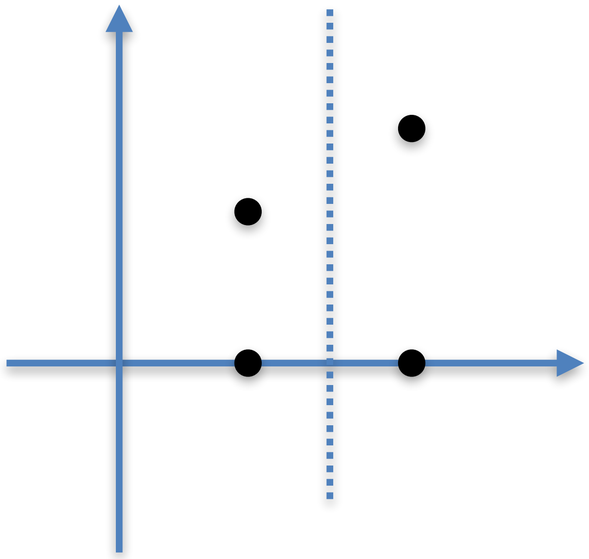
Decision boundaries must be parallel to one of the axes! They are defined by a threshold



Identification trees

Often numeric data is represented in this way

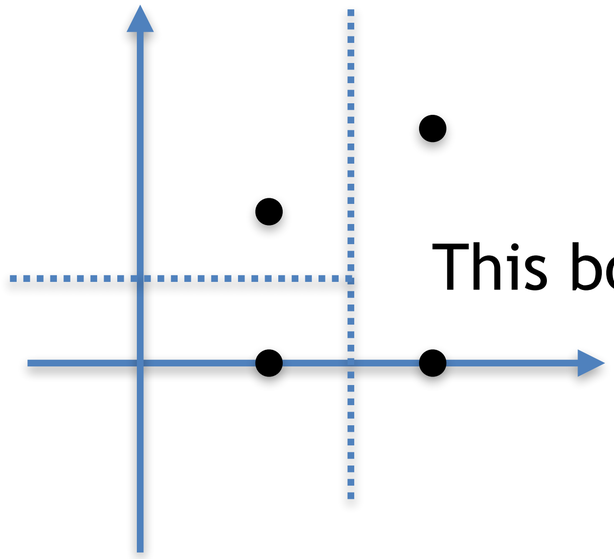
Decision boundaries must be parallel to one of the axes! They are defined by a threshold



Identification trees

Often numeric data is represented in this way

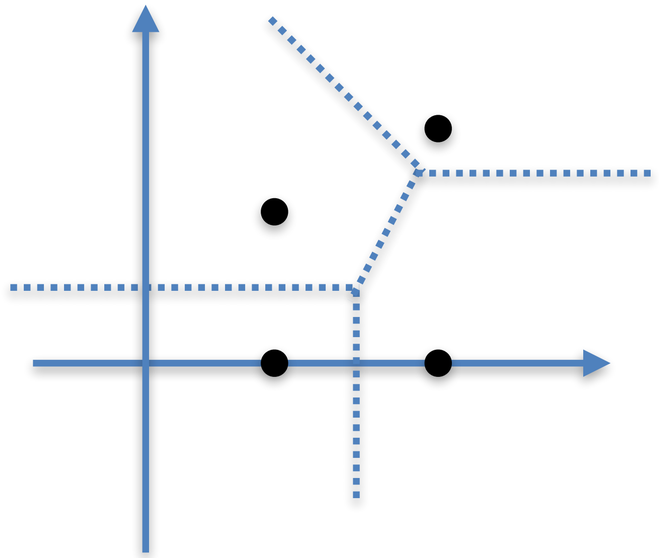
Decision boundaries must be parallel to one of the axes! They are defined by a threshold



This boundary does not extend here!

Identification trees

Nearest neighbors could give us this



Random forests

Trees are, as we saw, very simple and intuitive

Random forests

Trees are, as we saw, very simple and intuitive

In their simplest form however they are inaccurate

Random forests

Trees are, as we saw, very simple and intuitive

In their simplest form however they are inaccurate

They might be good in the training set, but they are not as good in validation

Random forests

Trees are, as we saw, very simple and intuitive

In their simplest form however they are inaccurate

They might be good in the training set, but they are not as good in validation

A possible solution (and there are many) is using so-called **random forests**!

Random forests

Trees are, as we saw, very simple and intuitive

In their simplest form however they are inaccurate

They might be good in the training set, but they are not as good in validation

A possible solution (and there are many) is using so-called **random forests**!

Random forests are one of the many **ensemble methods**

Random forests

The main idea of random forests is very similar to boostrapping for regressions

Random forests

The main idea of random forests is very similar to boostrapping for regressions

They allow “sampling” data and many different identification trees by considering a random number of features

Random forests

The main idea of random forests is very similar to boostrapping for regressions

They allow “sampling” data and many different identification trees by considering a random number of features

The classification is done considering a majority vote across many different trees

Random forests

First step to build a random forest is to get a boostrapped dataset

Random forests

First step to build a random forest is to get a boostrapped dataset

This is the same things we did for the regression case!

Random forests

First step to build a random forest is to get a boostrapped dataset

This is the same things we did for the regression case!

Pick s samples at random with replacement from the original dataset

Random forests

Shadow	Garlic	Complexion	Accent	Vampire
?	Yes	Pale	None	No
Yes	Yes	Ruddy	None	No
?	No	Ruddy	None	Yes
No	No	Average	Heavy	Yes
?	No	Average	Odd	Yes
Yes	No	Pale	Heavy	No
Yes	No	Average	Heavy	No
?	Yes	Ruddy	Odd	No

Original dataset

Random forests

Shadow	Garlic	Complexion	Accent	Vampire
?	Yes	Pale	None	No
Yes	Yes	Ruddy	None	No
?	No	Ruddy	None	Yes
No	No	Average	Heavy	Yes
?	No	Average	Odd	Yes
Yes	No	Pale	Heavy	No
Yes	No	Average	Heavy	No
?	Yes	Ruddy	Odd	No

Original dataset

Random forests

Shadow	Garlic	Complexion	Accent	Vampire
?	Yes	Pale	None	No
Yes	Yes	Ruddy	None	No
?	No	Ruddy	None	Yes
No	No	Average	Heavy	Yes
?	No	Average	Odd	Yes
Yes	No	Pale	Heavy	No
Yes	No	Average	Heavy	No
?	Yes	Ruddy	Odd	No

Shadow	Garlic	Complexion	Accent	Vampire
No	No	Average	Heavy	Yes
?	No	Ruddy	None	Yes
?	No	Ruddy	None	Yes
No	No	Average	Heavy	Yes
?	No	Average	Odd	Yes
Yes	No	Pale	Heavy	No
Yes	No	Pale	Heavy	No
Yes	No	Pale	Heavy	No

Original dataset

Boostrapped dataset

Random forests

Same sample are repeated!

Shadow	Garlic	Complexion	Accent	Vampire
?	Yes	Pale	None	No
Yes	Yes	Ruddy	None	No
?	No	Ruddy	None	Yes
No	No	Average	Heavy	Yes
?	No	Average	Odd	Yes
Yes	No	Pale	Heavy	No
Yes	No	Average	Heavy	No
?	Yes	Ruddy	Odd	No

Shadow	Garlic	Complexion	Accent	Vampire
No	No	Average	Heavy	Yes
?	No	Ruddy	None	Yes
?	No	Ruddy	None	Yes
No	No	Average	Heavy	Yes
?	No	Average	Odd	Yes
Yes	No	Pale	Heavy	No
Yes	No	Pale	Heavy	No
Yes	No	Pale	Heavy	No

Original dataset

Boostrapped dataset

Random forests

Second step is to create decision trees from the bootstrapped dataset

Random forests

Second step is to create decision trees from the bootstrapped dataset

Key aspect: only a random number of variables are used!

Random forests

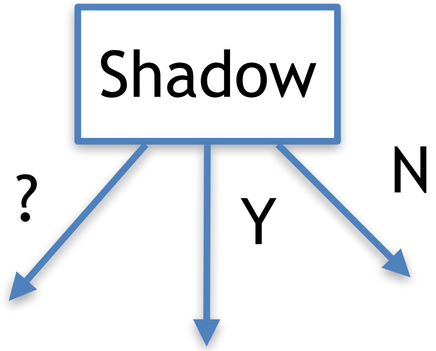
We could pick these two

Shadow	Garlic	Complexion	Accent	Vampire
No	No	Average	Heavy	Yes
?	No	Ruddy	None	Yes
?	No	Ruddy	None	Yes
No	No	Average	Heavy	Yes
?	No	Average	Odd	Yes
Yes	No	Pale	Heavy	No
Yes	No	Pale	Heavy	No
Yes	No	Pale	Heavy	No

Random forests

Imagine that “shadow” is the best, between the two, to split the sample

So, it will be the root of our tree

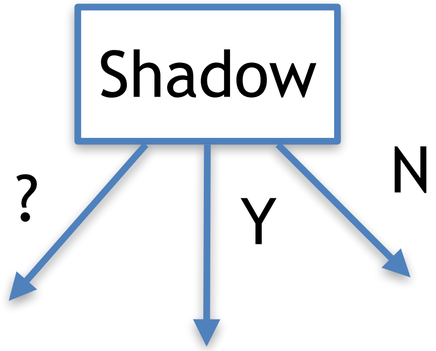


Shadow	Garlic	Complexion	Accent	Vampire
No	No	Average	Heavy	Yes
?	No	Ruddy	None	Yes
?	No	Ruddy	None	Yes
No	No	Average	Heavy	Yes
?	No	Average	Odd	Yes
Yes	No	Pale	Heavy	No
Yes	No	Pale	Heavy	No
Yes	No	Pale	Heavy	No

Random forests

Imagine that “shadow” is the best, between the two, to split the sample

So, it will be the root of our tree

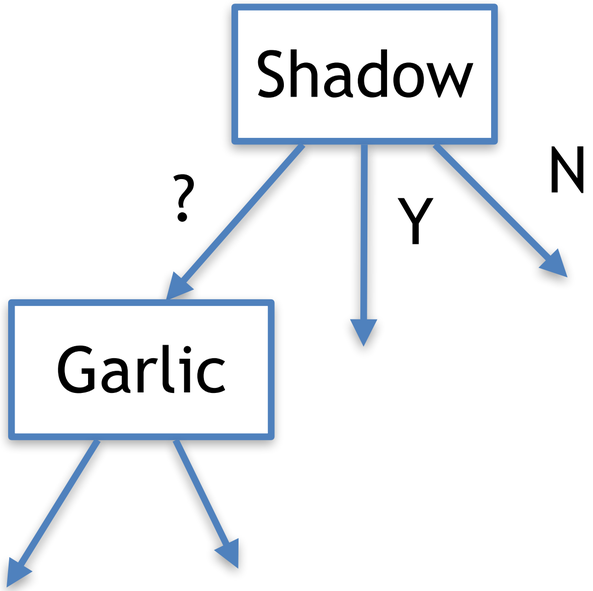


Then we keep building the tree by selecting again at random two other variables

Shadow	Garlic	Complexion	Accent	Vampire
No	No	Average	Heavy	Yes
?	No	Ruddy	None	Yes
?	No	Ruddy	None	Yes
No	No	Average	Heavy	Yes
?	No	Average	Odd	Yes
Yes	No	Pale	Heavy	No
Yes	No	Pale	Heavy	No
Yes	No	Pale	Heavy	No

Random forests

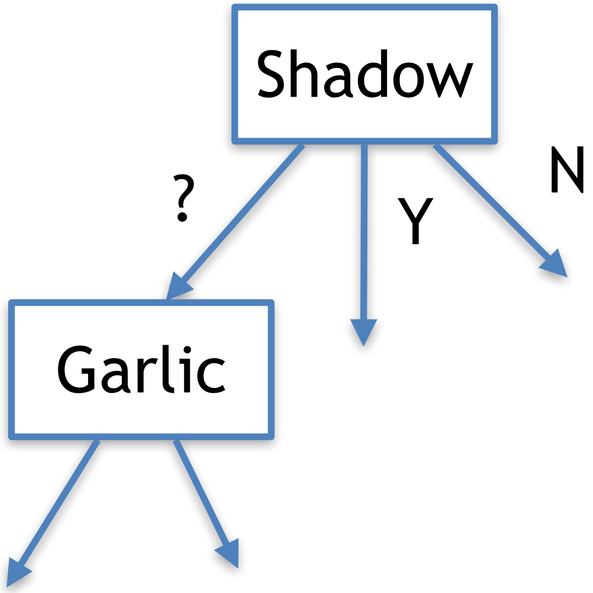
So we keep building the tree



Shadow	Garlic	Complexion	Accent	Vampire
No	No	Average	Heavy	Yes
?	No	Ruddy	None	Yes
?	No	Ruddy	None	Yes
No	No	Average	Heavy	Yes
?	No	Average	Odd	Yes
Yes	No	Pale	Heavy	No
Yes	No	Pale	Heavy	No
Yes	No	Pale	Heavy	No

Random forests

So we keep building the tree

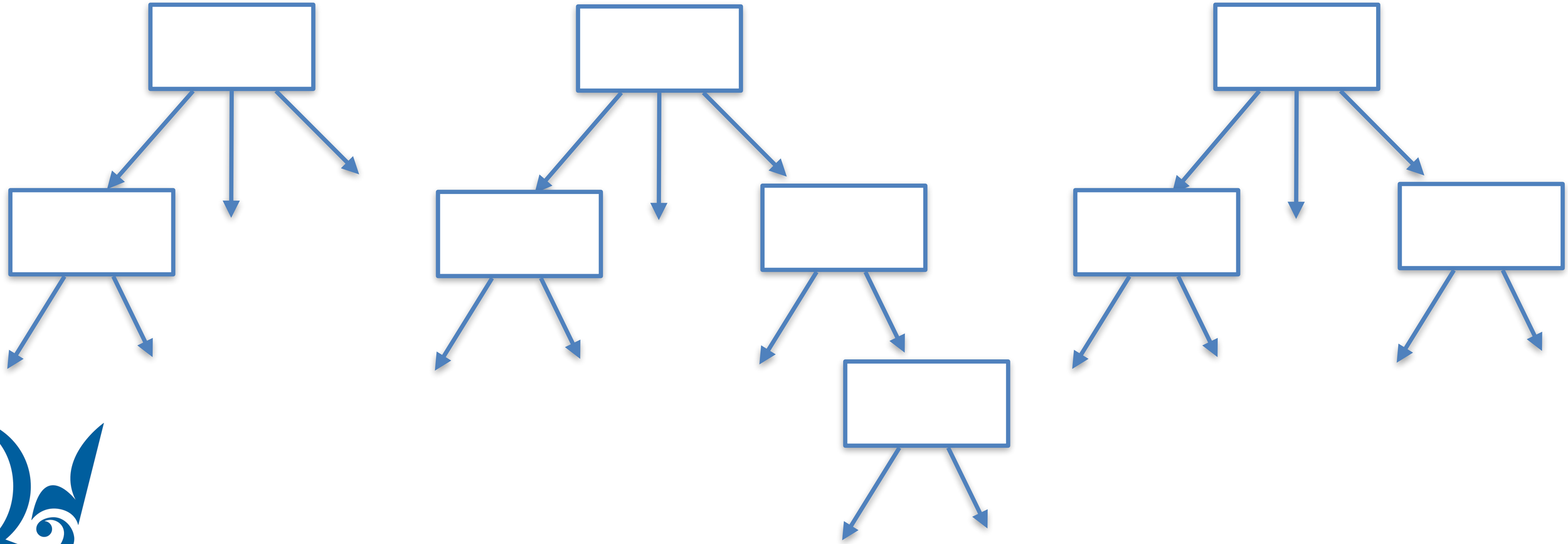


Shadow	Garlic	Complexion	Accent	Vampire
No	No	Average	Heavy	Yes
?	No	Ruddy	None	Yes
?	No	Ruddy	None	Yes
No	No	Average	Heavy	Yes
?	No	Average	Odd	Yes
Yes	No	Pale	Heavy	No
Yes	No	Pale	Heavy	No
Yes	No	Pale	Heavy	No

We keep going as usual until we have found all the leafs

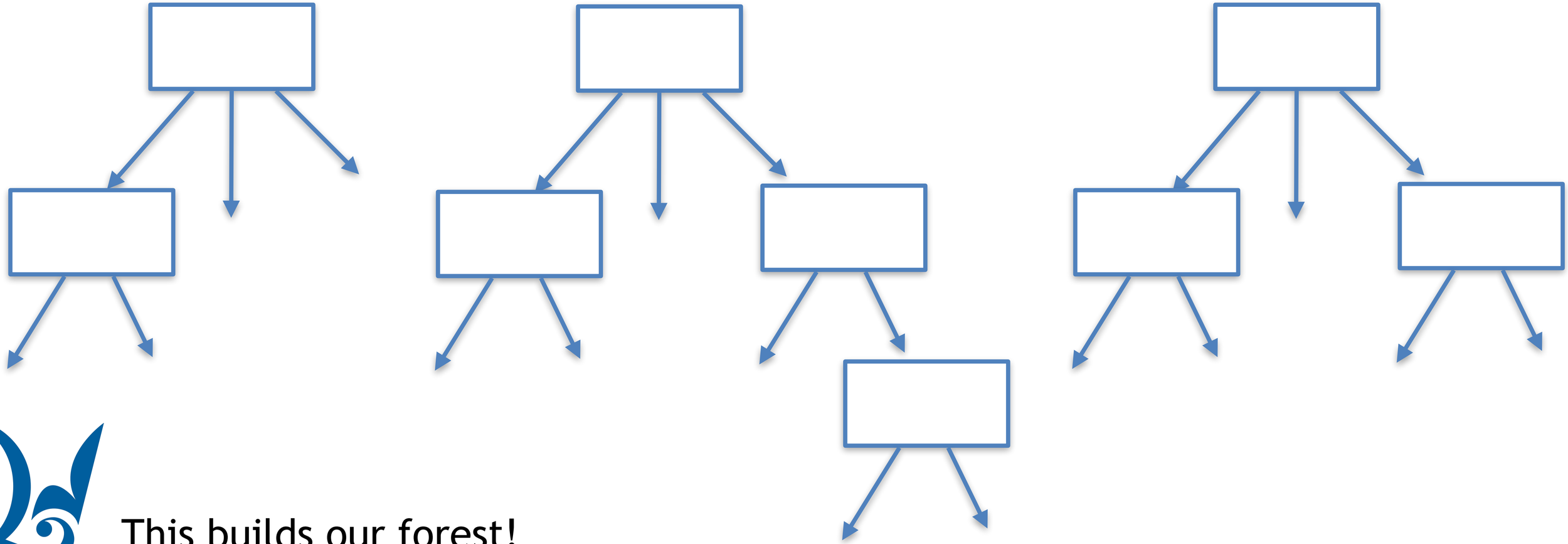
Random forests

Then we go back to step 1 and repeat and get another tree many times



Random forests

Then we go back to step 1 and repeat and get another tree many times



This builds our forest!

Random forests

The variety that of decision trees that we build it turns out to improve their precision

Random forests

The variety that of decision trees that we build it turns out to improve their precision

But, how do we use it for real?

Random forests

It is very simple, we get a new data point that we need to classify

Random forests

It is very simple, we get a new data point that we need to classify

Shadow	Garlic	Complexion	Accent	Vampire
No	No	Average	Heavy	?

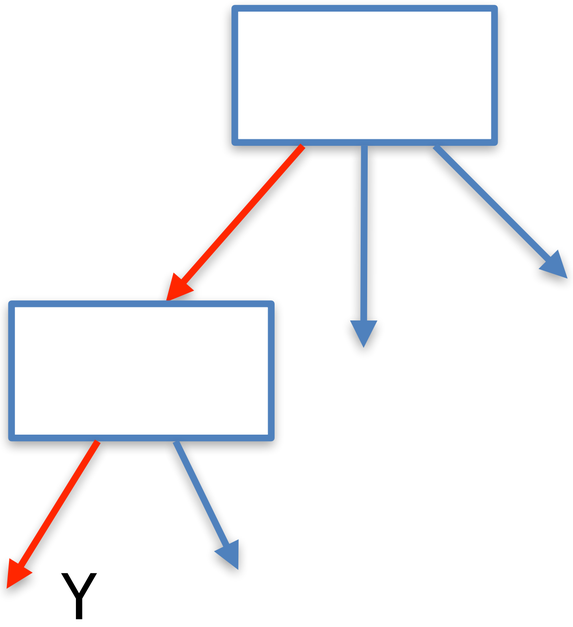
Random forests

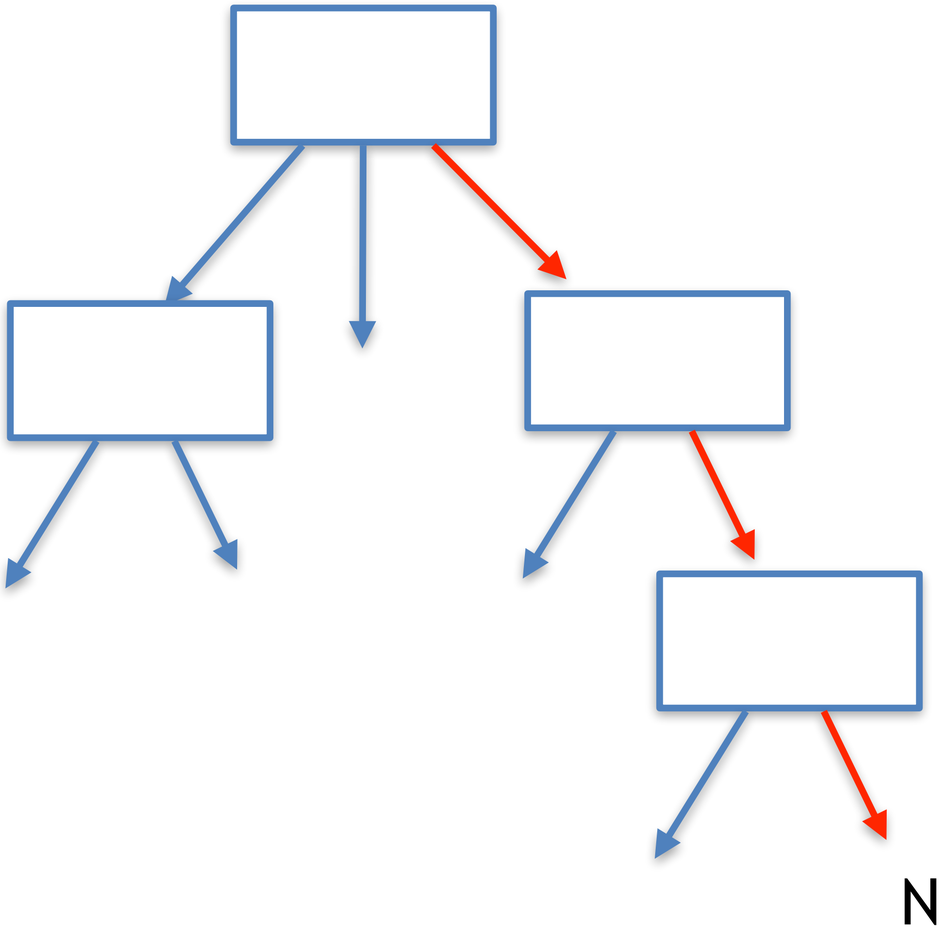
It is very simple, we get a new data point that we need to classify

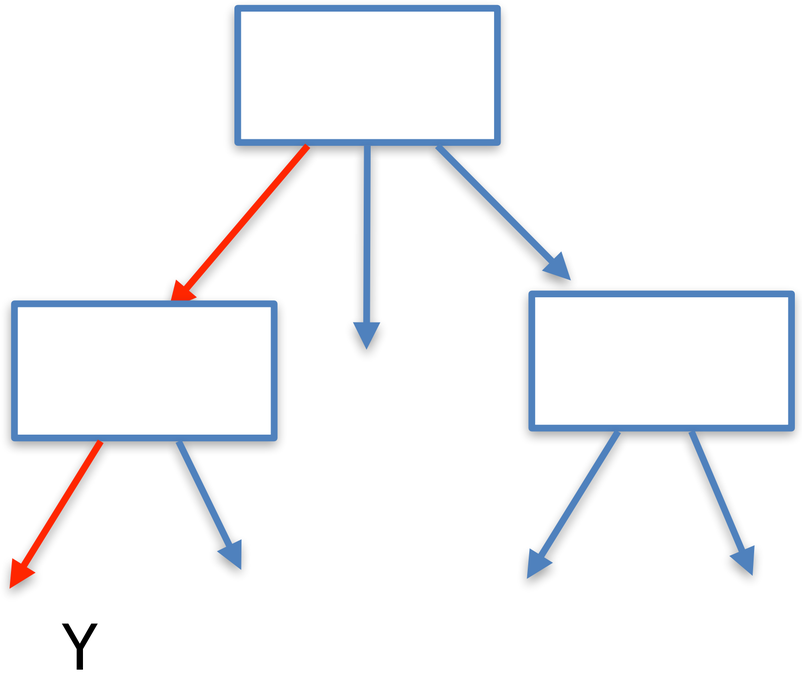
Shadow	Garlic	Complexion	Accent	Vampire
No	No	Average	Heavy	?

Then we run this data point in each decision tree in the forest

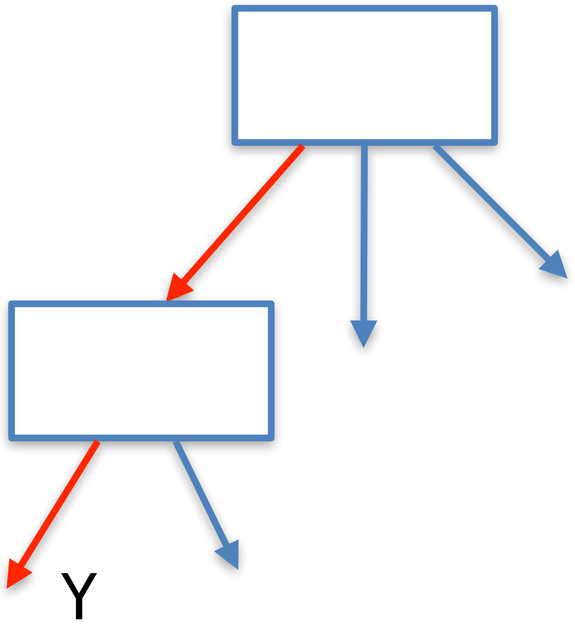
Random forests

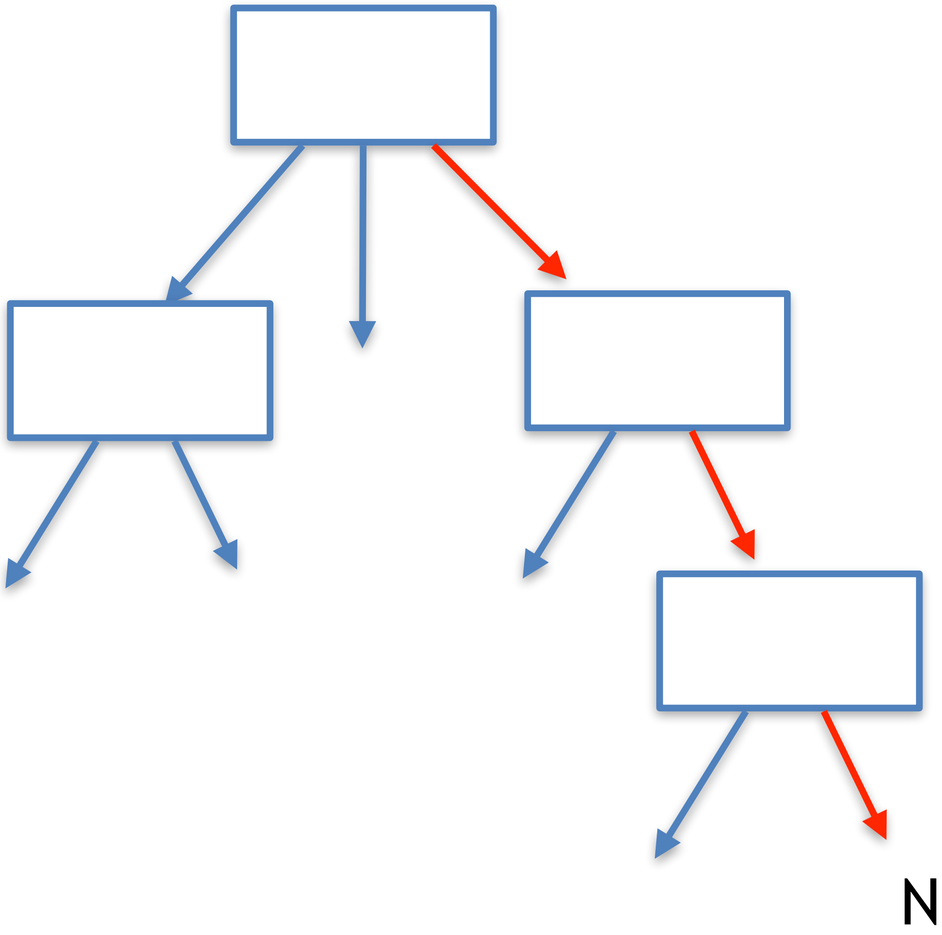


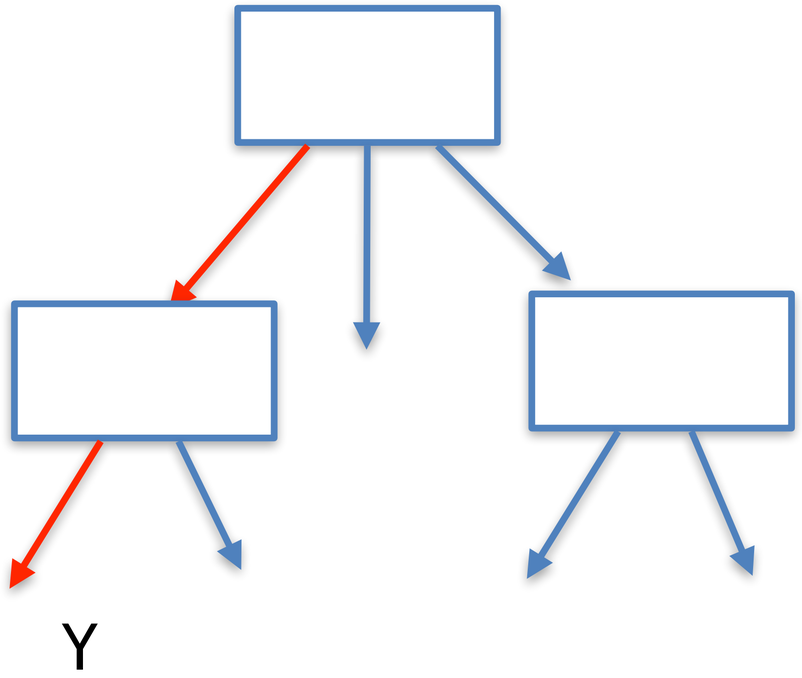




Random forests

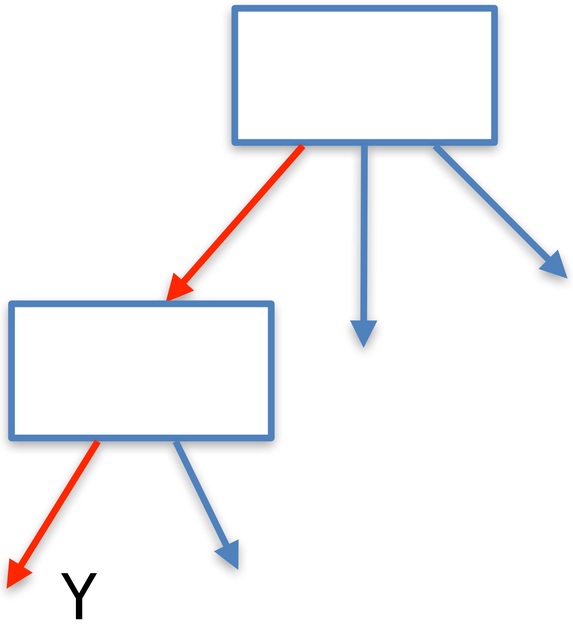


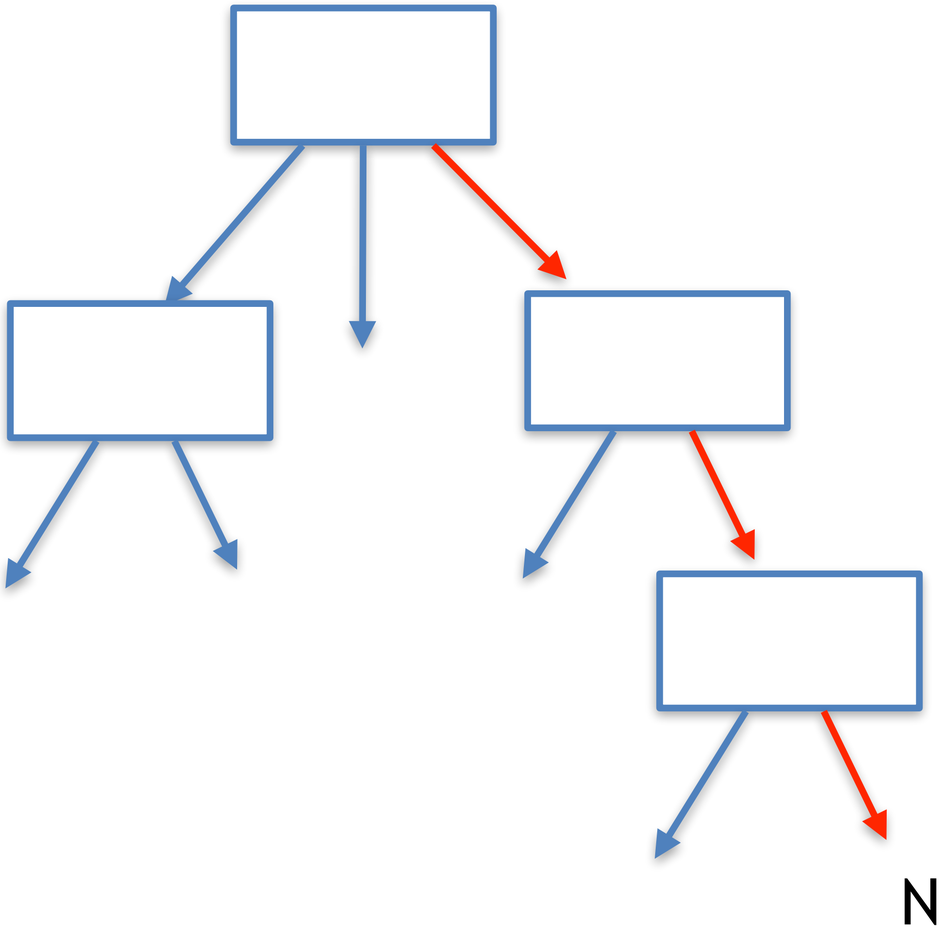


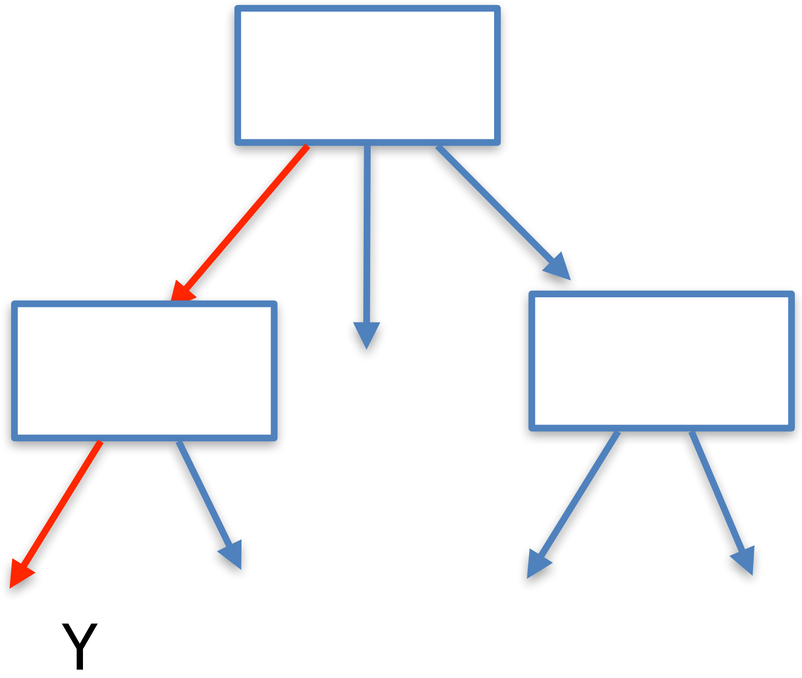


In this simple case, the new data point is classified as a vampire 2 times over 3

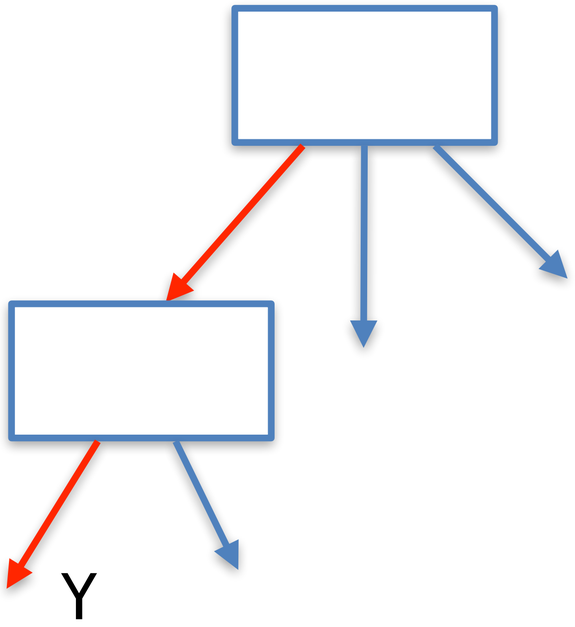
Random forests

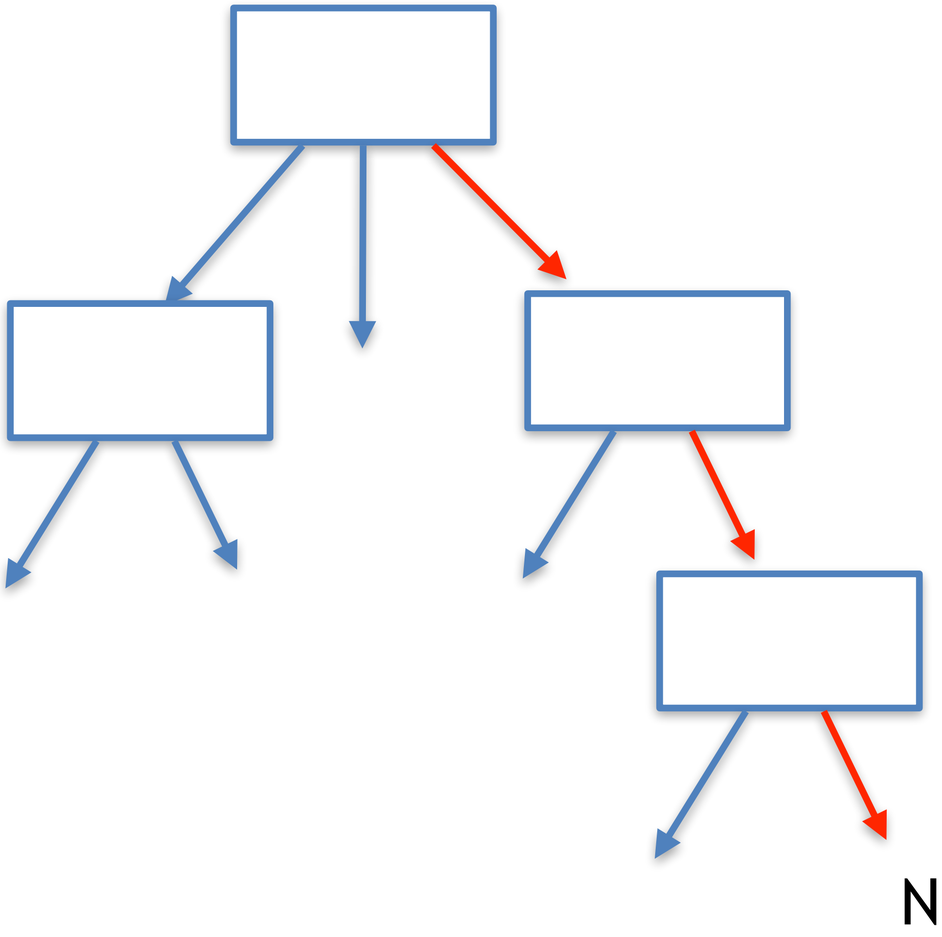


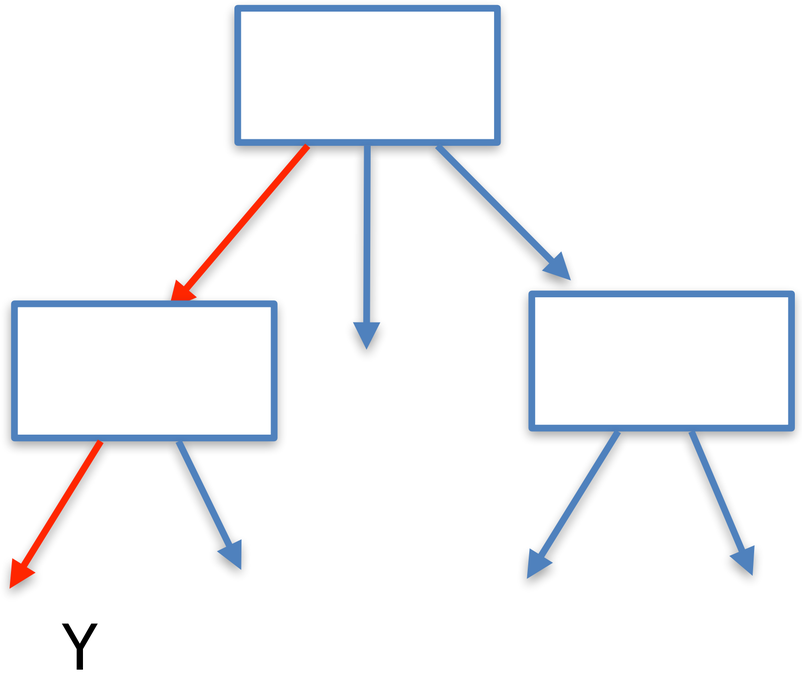




Random forests







The option with the majority of “votes” is the solution!

Random forests

Key question: how do we know or estimate the precision of the forest?

Random forests

In the boostrapped dataset, how many samples are not selected, on average?

Random forests

In the boostrapped dataset, how many samples are not selected, on average?

Probability of being selected in one draw is $\frac{1}{s}$

Random forests

In the boostrapped dataset, how many samples are not selected, on average?

Probability of being selected in one draw is $\frac{1}{s}$

Probability of NOT being selected in one draw is $1 - \frac{1}{s}$

Random forests

In the boostrapped dataset, how many samples are not selected, on average?

Probability of being selected in one draw is $\frac{1}{s}$

Probability of NOT being selected in one draw is $1 - \frac{1}{s}$

Probability of NOT being selected after s draws is $\left(1 - \frac{1}{s}\right)^{s} \xrightarrow{s \to \infty} \frac{1}{e}$

Random forests

In the boostrapped dataset, how many samples are not selected, on average?

Probability of being selected in one draw is $\frac{1}{s}$

Probability of NOT being selected in one draw is $1 - \frac{1}{s}$

Probability of NOT being selected after s draws is $\left(1 - \frac{1}{s}\right)^{s} \xrightarrow{s \to \infty} \frac{1}{e}$

So, we can use this 36% of the data for evaluation!

Random forests

In the boostrapped dataset, how many samples are not selected, on average?

Probability of being selected in one draw is $\frac{1}{s}$

Probability of NOT being selected in one draw is $1 - \frac{1}{s}$

Probability of NOT being selected after s draws is $\left(1 - \frac{1}{s}\right)^{s} \xrightarrow{s \to \infty} \frac{1}{e}$

So, we can use this 36% of the data for evaluation!

This subsample is called out-of-bag sample

Random forests

However, how do we pick the number of random variables to build the forest?

Random forests

However, how do we pick the number of random variables to build the forest?

We use this number of hyper-parameter and confront the out-of-bag error

Random forests

However, how do we pick the number of random variables to build the forest?

We use this number of hyper-parameter and confront the out-of-bag error

The starting point is often the square-root of the number of variables, then you consider values below and above this

AdaBoost (adaptive boosting)

So far, we have used either one or multiple “full” decision trees

AdaBoost (adaptive boosting)

So far, we have used either one or multiple “full” decision trees

It turns out that we can use many weak leaners (i.e., stumps) instead

AdaBoost (adaptive boosting)

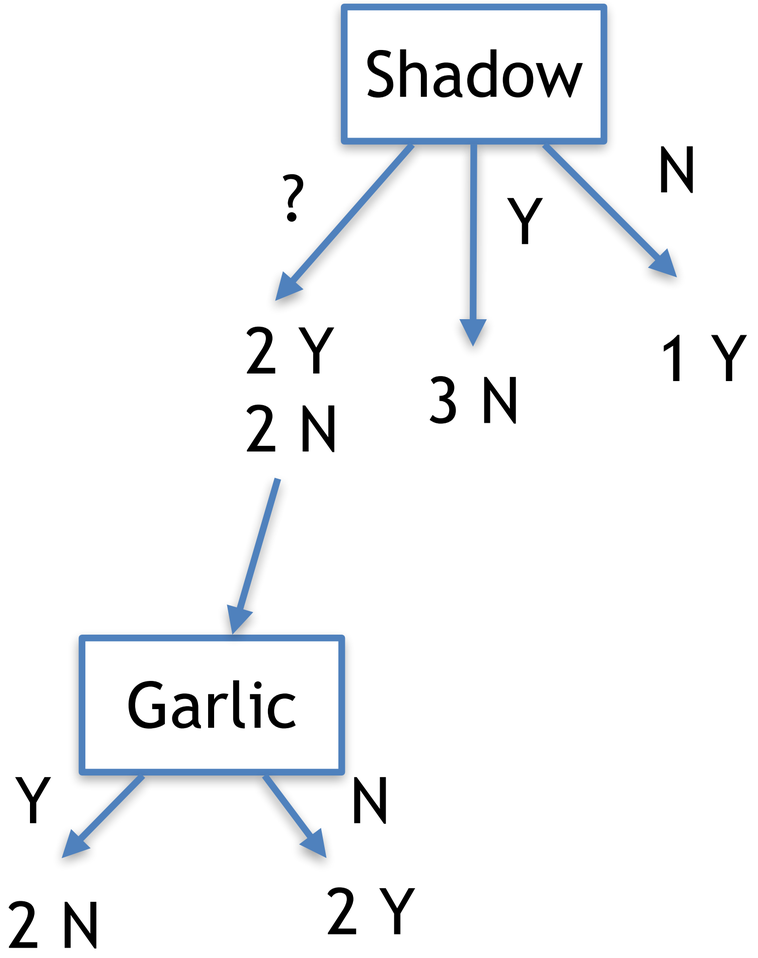
So far, we have used either one or multiple “full” decision trees

It turns out that we can use many weak leaners (i.e., stumps) instead

We use a forest of stumps

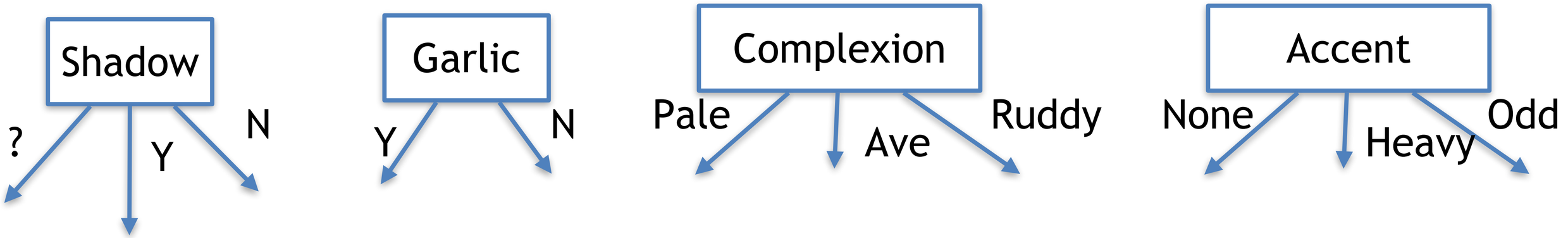
AdaBoost

This was our first full tree



AdaBoost

In AdaBoost we will be using stumps



AdaBoost VS random forest

In random forests, each tree had an equal weight on the final decision

AdaBoost VS random forest

In random forests, each tree had an equal weight on the final decision

In a forest of stumps made with AdaBoost, some will be considered as more important than others

AdaBoost VS random forest

In random forests, each tree had an equal weight on the final decision

In a forest of stumps made with AdaBoost, some will be considered as more important than others

In random forests, each tree was independent of the others

AdaBoost VS random forest

In random forests, each tree had an equal weight on the final decision

In a forest of stumps made with AdaBoost, some will be considered as more important than others

In random forests, each tree was independent of the others

In a forest of stumps made with AdaBoost, order is key

AdaBoost VS random forest

In random forests, each tree had an equal weight on the final decision

In a forest of stumps made with AdaBoost, some will be considered as more important than others

In random forests, each tree was independent of the others

In a forest of stumps made with AdaBoost, order is key

In particular, the error that the first stump makes influence how the second stump is made, the error of the second stump influence the third etc..

AdaBoost

Let us now see how this is done

AdaBoost

Let us now see how this is done

We start adding a weight defining how important is to correctly classify each sample

AdaBoost

Let us now see how this is done

We start adding a weight defining how important is to correctly classify each sample

Shadow	Garlic	Complexion	Accent	Vampire	Weight
?	Yes	Pale	None	No	1/8
Yes	Yes	Ruddy	None	No	1/8
?	No	Ruddy	None	Yes	1/8
No	No	Average	Heavy	Yes	1/8
?	No	Average	Odd	Yes	1/8
Yes	No	Pale	Heavy	No	1/8
Yes	No	Average	Heavy	No	1/8
?	Yes	Ruddy	Odd	No	1/8

AdaBoost

Let us now see how this is done

We start adding a weight defining how important is to correctly classify each sample

Shadow	Garlic	Complexion	Accent	Vampire	Weight
?	Yes	Pale	None	No	1/8
Yes	Yes	Ruddy	None	No	1/8
?	No	Ruddy	None	Yes	1/8
No	No	Average	Heavy	Yes	1/8
?	No	Average	Odd	Yes	1/8
Yes	No	Pale	Heavy	No	1/8
Yes	No	Average	Heavy	No	1/8
?	Yes	Ruddy	Odd	No	1/8

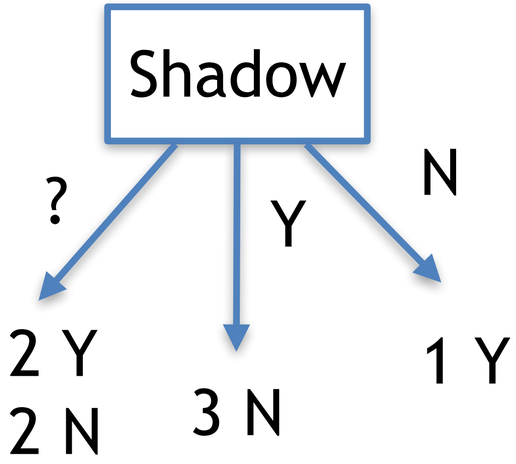
At the start each weight is the same: $\frac{1}{s}$

AdaBoost

We then evaluate which stump does the best job classifying the data using a majority rule

AdaBoost

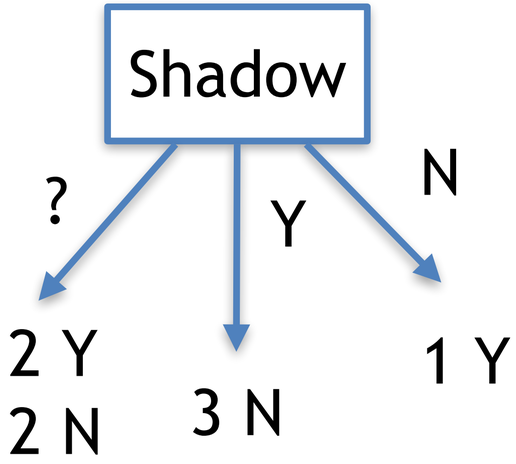
We then evaluate which stump does the best job classifying the data using a majority rule



Shadow	Garlic	Complexion	Accent	Vampire	Weight
?	Yes	Pale	None	No	1/8
Yes	Yes	Ruddy	None	No	1/8
?	No	Ruddy	None	Yes	1/8
No	No	Average	Heavy	Yes	1/8
?	No	Average	Odd	Yes	1/8
Yes	No	Pale	Heavy	No	1/8
Yes	No	Average	Heavy	No	1/8
?	Yes	Ruddy	Odd	No	1/8

AdaBoost

We then evaluate which stump does the best job classifying the data using a majority rule

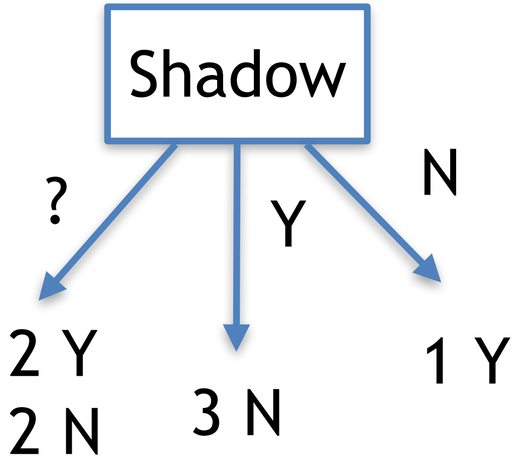


Classification

Shadow	Garlic	Complexion	Accent	Vampire	Weight
?	Yes	Pale	None	No	1/8
Yes	Yes	Ruddy	None	No	1/8
?	No	Ruddy	None	Yes	1/8
No	No	Average	Heavy	Yes	1/8
?	No	Average	Odd	Yes	1/8
Yes	No	Pale	Heavy	No	1/8
Yes	No	Average	Heavy	No	1/8
?	Yes	Ruddy	Odd	No	1/8

AdaBoost

We then evaluate which stump does the best job classifying the data using a majority rule



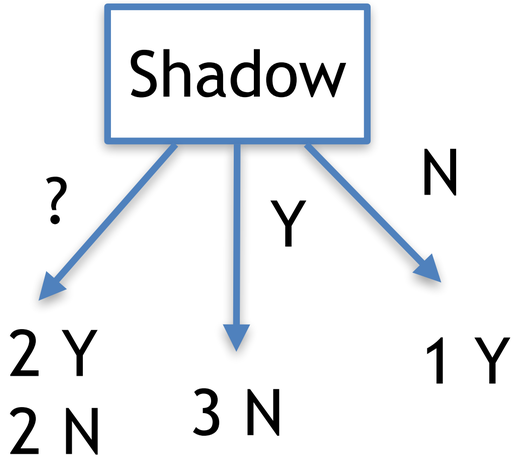
Classification

If no -> yes

Shadow	Garlic	Complexion	Accent	Vampire	Weight
?	Yes	Pale	None	No	1/8
Yes	Yes	Ruddy	None	No	1/8
?	No	Ruddy	None	Yes	1/8
No	No	Average	Heavy	Yes	1/8
?	No	Average	Odd	Yes	1/8
Yes	No	Pale	Heavy	No	1/8
Yes	No	Average	Heavy	No	1/8
?	Yes	Ruddy	Odd	No	1/8

AdaBoost

We then evaluate which stump does the best job classifying the data using a majority rule



Classification

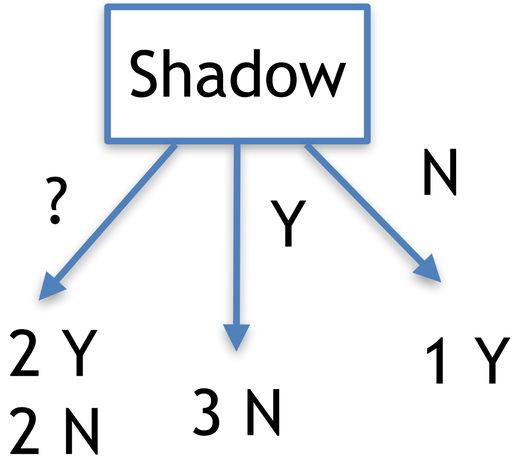
If no -> yes

If yes -> no

Shadow	Garlic	Complexion	Accent	Vampire	Weight
?	Yes	Pale	None	No	1/8
Yes	Yes	Ruddy	None	No	1/8
?	No	Ruddy	None	Yes	1/8
No	No	Average	Heavy	Yes	1/8
?	No	Average	Odd	Yes	1/8
Yes	No	Pale	Heavy	No	1/8
Yes	No	Average	Heavy	No	1/8
?	Yes	Ruddy	Odd	No	1/8

AdaBoost

We then evaluate which stump does the best job classifying the data using a majority rule



Classification

If no -> yes

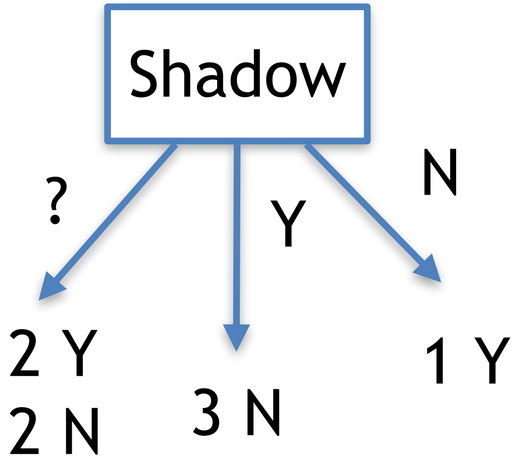
If yes -> no

If ? -> yes (or no it is the same!)

Shadow	Garlic	Complexion	Accent	Vampire	Weight
?	Yes	Pale	None	No	1/8
Yes	Yes	Ruddy	None	No	1/8
?	No	Ruddy	None	Yes	1/8
No	No	Average	Heavy	Yes	1/8
?	No	Average	Odd	Yes	1/8
Yes	No	Pale	Heavy	No	1/8
Yes	No	Average	Heavy	No	1/8
?	Yes	Ruddy	Odd	No	1/8

AdaBoost

We then evaluate which stump does the best job classifying the data using a majority rule



Classification

If no -> yes

If yes -> no

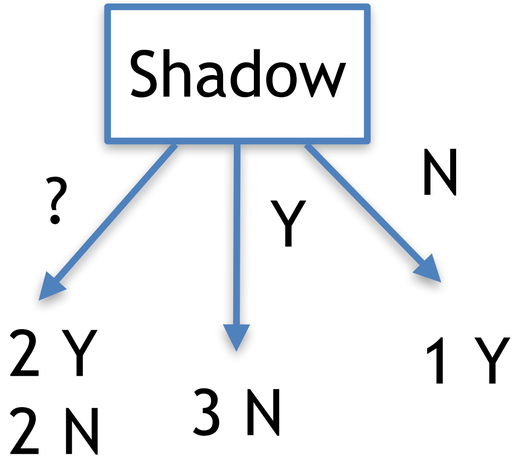
2 mistakes

If ? -> yes (or no it is the same!)

Shadow	Garlic	Complexion	Accent	Vampire	Weight
?	Yes	Pale	None	No	1/8
Yes	Yes	Ruddy	None	No	1/8
?	No	Ruddy	None	Yes	1/8
No	No	Average	Heavy	Yes	1/8
?	No	Average	Odd	Yes	1/8
Yes	No	Pale	Heavy	No	1/8
Yes	No	Average	Heavy	No	1/8
?	Yes	Ruddy	Odd	No	1/8

AdaBoost

We then evaluate which stump does the best job classifying the data using a majority rule



Classification

If no -> yes

If yes -> no

2 mistakes

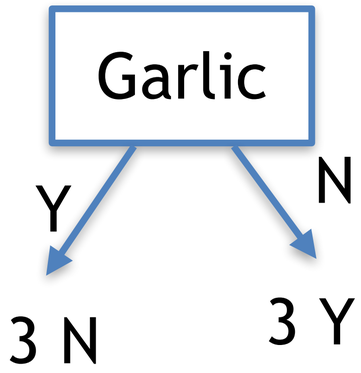
6 correct

If ? -> yes (or no it is the same!)

Shadow	Garlic	Complexion	Accent	Vampire	Weight
?	Yes	Pale	None	No	1/8
Yes	Yes	Ruddy	None	No	1/8
?	No	Ruddy	None	Yes	1/8
No	No	Average	Heavy	Yes	1/8
?	No	Average	Odd	Yes	1/8
Yes	No	Pale	Heavy	No	1/8
Yes	No	Average	Heavy	No	1/8
?	Yes	Ruddy	Odd	No	1/8

AdaBoost

We then which stump does the best job classifying the data



Classification

If yes -> no

If no -> yes

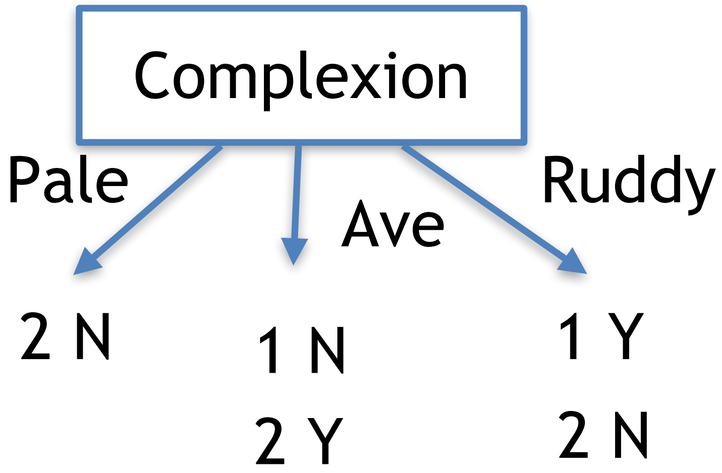
2 mistakes

6 correct

Shadow	Garlic	Complexion	Accent	Vampire	Weight
?	Yes	Pale	None	No	1/8
Yes	Yes	Ruddy	None	No	1/8
?	No	Ruddy	None	Yes	1/8
No	No	Average	Heavy	Yes	1/8
?	No	Average	Odd	Yes	1/8
Yes	No	Pale	Heavy	No	1/8
Yes	No	Average	Heavy	No	1/8
?	Yes	Ruddy	Odd	No	1/8

AdaBoost

We then which stump does the best job classifying the data



Classification

If Pale -> no

If Ave -> yes

If Ruddy -> no

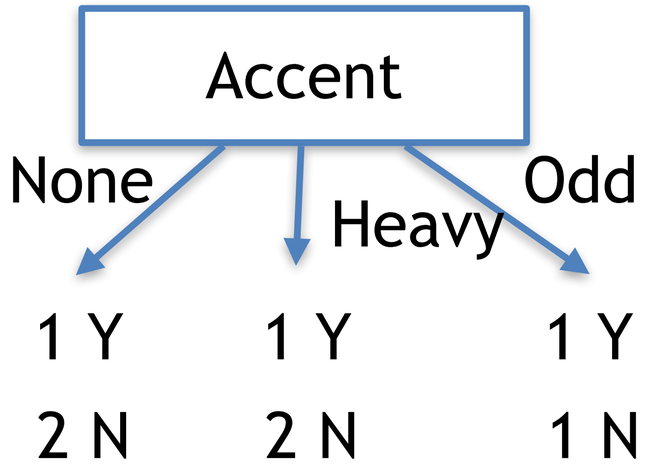
2 mistakes

6 correct

Shadow	Garlic	Complexion	Accent	Vampire	Weight
?	Yes	Pale	None	No	1/8
Yes	Yes	Ruddy	None	No	1/8
?	No	Ruddy	None	Yes	1/8
No	No	Average	Heavy	Yes	1/8
?	No	Average	Odd	Yes	1/8
Yes	No	Pale	Heavy	No	1/8
Yes	No	Average	Heavy	No	1/8
?	Yes	Ruddy	Odd	No	1/8

AdaBoost

We then which stump does the best job classifying the data



Classification

If None -> no

If Heavy -> no

If odd -> yes (or no it is the same!)

3 mistakes

5 correct

Shadow	Garlic	Complexion	Accent	Vampire	Weight
?	Yes	Pale	None	No	1/8
Yes	Yes	Ruddy	None	No	1/8
?	No	Ruddy	None	Yes	1/8
No	No	Average	Heavy	Yes	1/8
?	No	Average	Odd	Yes	1/8
Yes	No	Pale	Heavy	No	1/8
Yes	No	Average	Heavy	No	1/8
?	Yes	Ruddy	Odd	No	1/8

AdaBoost

We now select the best stump using, for example, the Gini Impurity

AdaBoost

We now select the best stump using, for example, the Gini Impurity

Shadow has the smaller GI, it will be our first stump

AdaBoost

We now select the best stump using, for example, the Gini Impurity

Shadow has the smaller GI, it will be our first stump

We now have to compute the weight of this stump based on the error it made in the classification

AdaBoost

We now select the best stump using, for example, the Gini Impurity

Shadow has the smaller GI, it will be our first stump

We now have to compute the weight of this stump based on the error it made in the classification

To do that we first calculate the total error of the stump as the sum of the weights of the wrongly classified data

AdaBoost

We now select the best stump using, for example, the Gini Impurity

Shadow has the smaller GI, it will be our first stump

We now have to compute the weight of this stump based on the error it made in the classification

To do that we first calculate the total error of the stump as the sum of the weights of the wrongly classified data

Since the stump made two mistakes and each data sample had the same weight t
total error is

AdaBoost

We now select the best stump using, for example, the Gini Impurity

Shadow has the smaller GI, it will be our first stump

We now have to compute the weight of this stump based on the error it made in the classification

To do that we first calculate the total error of the stump as the sum of the weights of the wrongly classified data

Since the stump made two mistakes and each data sample had the same weight t
total error is

$$E_t = \frac{1}{8} + \frac{1}{8} = \frac{1}{4}$$

AdaBoost

Using the total error of the stump we can calculate its weight as

AdaBoost

Using the total error of the stump we can calculate its weight as

$$w_{stump} = \frac{1}{2} \ln\left(\frac{1 - E_t}{E_t}\right) \sim 0.54$$

AdaBoost

Using this information we can modify the weights of the samples

AdaBoost

Using this information we can modify the weights of the samples

The idea is to give more weights to the samples that were not correctly classified

AdaBoost

Using this information we can modify the weights of the samples

The idea is to give more weights to the samples that were not correctly classified

This is done using this formula

AdaBoost

Using this information we can modify the weights of the samples

The idea is to give more weights to the samples that were not correctly classified

This is done using this formula

$$w_{s_i} = s_i e^{w_{stump}} = \frac{1}{8} e^{0.54} \sim 0.21$$

AdaBoost

Using this information we can modify the weights of the samples

The idea is to give more weights to the samples that were not correctly classified

This is done using this formula

$$w_{s_i} = s_i e^{w_{stump}} = \frac{1}{8} e^{0.54} \sim 0.21$$

We then decrease the sample weights for the correctly classified samples as

AdaBoost

Using this information we can modify the weights of the samples

The idea is to give more weights to the samples that were not correctly classified

This is done using this formula

$$w_{s_i} = s_i e^{w_{stump}} = \frac{1}{8} e^{0.54} \sim 0.21$$

We then decrease the sample weights for the correctly classified samples as

$$w_{s_i} = s_i e^{-w_{stump}} = \frac{1}{8} e^{-0.54} \sim 0.07$$

AdaBoost

Hence, we will have

Shadow	Garlic	Complexion	Accent	Vampire	Weight
?	Yes	Pale	None	No	0.21
Yes	Yes	Ruddy	None	No	0.07
?	No	Ruddy	None	Yes	0.07
No	No	Average	Heavy	Yes	0.07
?	No	Average	Odd	Yes	0.07
Yes	No	Pale	Heavy	No	0.07
Yes	No	Average	Heavy	No	0.07
?	Yes	Ruddy	Odd	No	0.21

We need to normalize the weight!

AdaBoost

Hence, we will have

Shadow	Garlic	Complexion	Accent	Vampire	Weight
?	Yes	Pale	None	No	0.25
Yes	Yes	Ruddy	None	No	0.083
?	No	Ruddy	None	Yes	0.083
No	No	Average	Heavy	Yes	0.083
?	No	Average	Odd	Yes	0.083
Yes	No	Pale	Heavy	No	0.083
Yes	No	Average	Heavy	No	0.083
?	Yes	Ruddy	Odd	No	0.25

AdaBoost

Now we can create a boostrapped version of the sample, using the weights to sampling

AdaBoost

Now we can create a boostrapped version of the sample, using the weights to sampling

This will create a dataset with over-represented samples that were not correctly Classified. Hence, the error in the initial step affect the stump in the second

AdaBoost

Now we can create a boostrapped version of the sample, using the weights to sampling

This will create a dataset with over-represented samples that were not correctly Classified. Hence, the error in the initial step affect the stump in the second

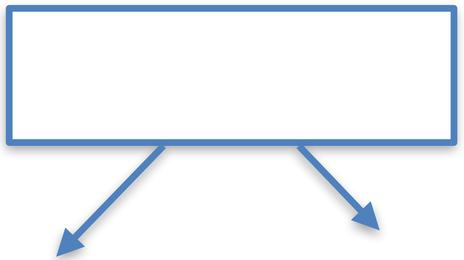
Then, we forget about the initial dataset, we set all the sample weights to zero and repeat

AdaBoost

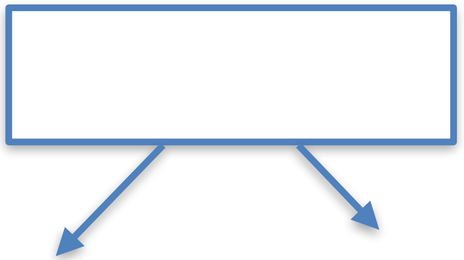
Hence, we will have a range of stumps, each with their weight

AdaBoost

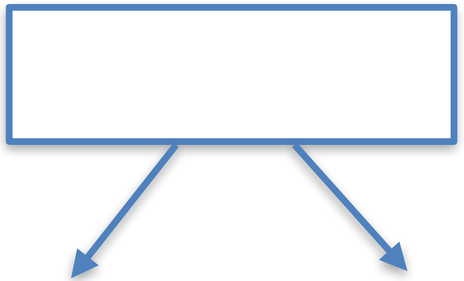
For example



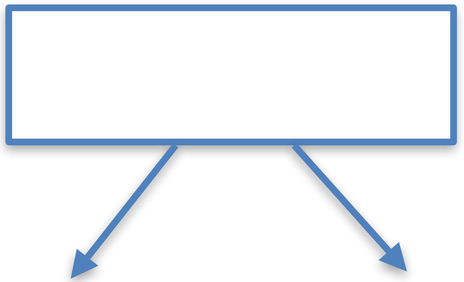
$w_{stump} = 0.54$



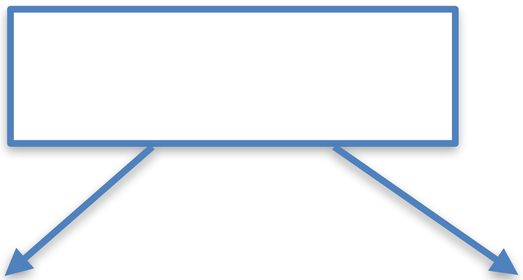
$w_{stump} = 0.4$



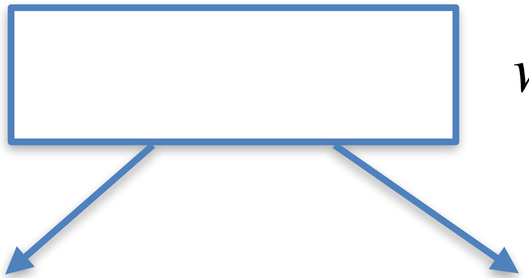
$w_{stump} = 0.6$



$w_{stump} = 0.34$



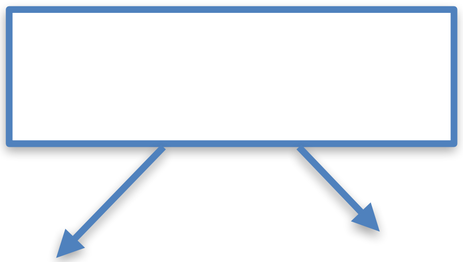
$w_{stump} = 0.3$



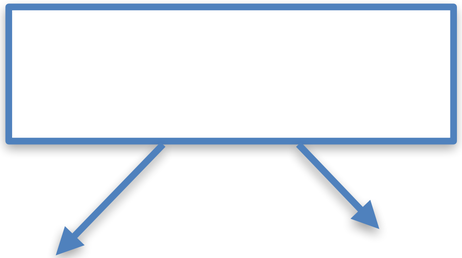
$w_{stump} = 0.1$

AdaBoost

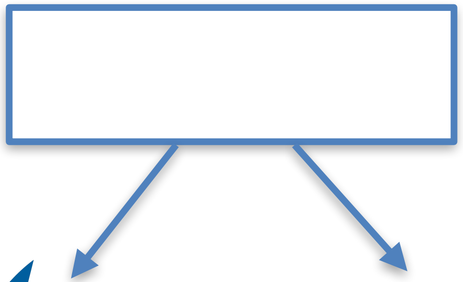
Imagine we get a new data point, stumps on the left classify it as vampire, those on the left as not a vampire. We pick the classification by summing the weights of each stump!



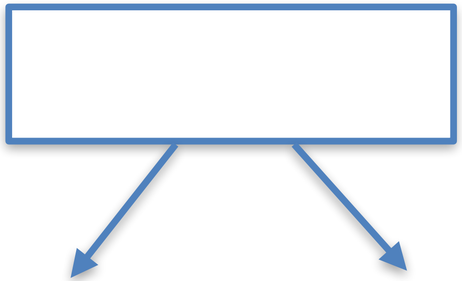
$w_{stump} = 0.54$



$w_{stump} = 0.4$



$w_{stump} = 0.6$



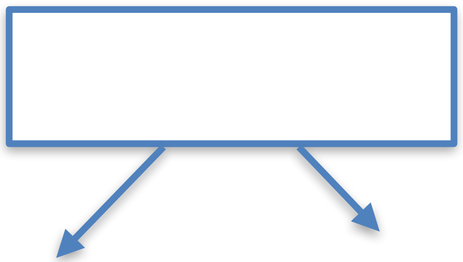
$w_{stump} = 0.34$

$w_{stump} = 0.3$

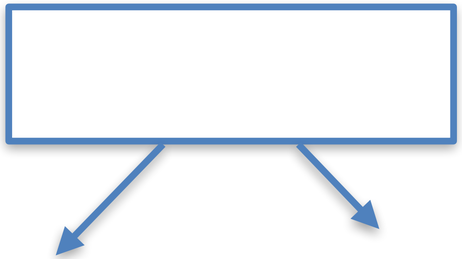
$w_{stump} = 0.1$

AdaBoost

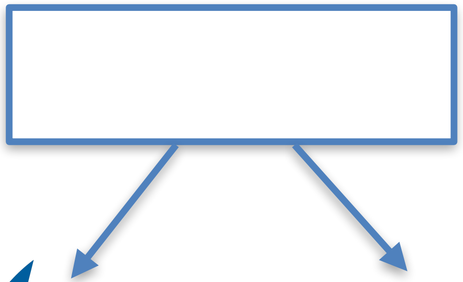
So, in this case, the data point will be classified as a vampire



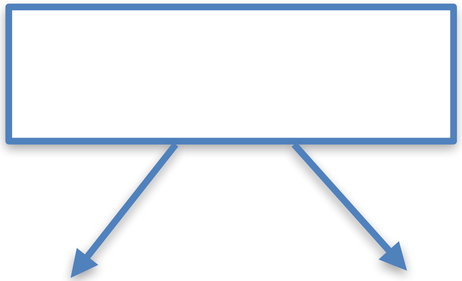
$w_{stump} = 0.54$



$w_{stump} = 0.4$



$w_{stump} = 0.6$



$w_{stump} = 0.34$

$w_{stump} = 0.3$

$w_{stump} = 0.1$