We start with a dataset which we want to classify.

Ear shape	Face shape	Whiskers	Cat
Pointy	Round	Present	Yes
Floppy	Round	Absent	No
Floppy	Round	Absent	No
Pointy	Round	Present	Yes
Pointy	Not Round	Present	Yes
Floppy	Round	Absent	No
Floppy	Round	Present	Yes
Pointy	Not Round	Absent	No
Pointy	Not Round	Absent	No
Pointy	Not Round	Present	Yes

Since we can't use the same training set again and again and expect a different tree, we will use bootstrap sampling.

Bootstrap sampling is a technique used by random forest algorithm. Basically, how we find probability with replacement.

EXAMPLE 3 Joanie and Chachi are friends. Each has a box that contains 3 raisin cookies and 5 lemon cookies.

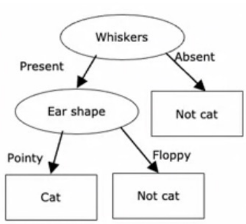
A Without looking, Joanie takes a cookie from her box. This first cookie is lemon, so she puts it back, mixes up the cookies, and picks another cookie. The second cookie is lemon as well. What is the probability of Joanie's 2-lemon-cookie outcome (which was with replacement)?

$P(1^{st}\text{ lemon}) = \frac{5}{8}$

$P(2^{nd}\text{ lemon}) = \frac{5}{8}$

$P(\text{both lemon}) = \frac{5}{8}\cdot\frac{5}{8} = \frac{25}{64}$

The original dataset will give this decision tree :-



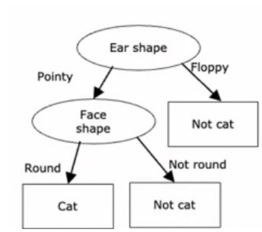
We will apply bootstrap sampling to obtain a new dataset.

Since we are selecting random elements from our original dataset "with replacement" there's a high chance that some element gets repeated and that repetition makes us obtain a new training set from the old one

New dataset →

Ear shape	Face shape	Whiskers	Cat
Pointy	Round	Present	Yes
Pointy	Round	Absent	Yes
Floppy	Not Round	Absent	No
Floppy	Not Round	Absent	No
Pointy	Round	Absent	Yes
Floppy	Round	Absent	No
Floppy	Round	Absent	No
Floppy	Round	Absent	No
Pointy	Not Round	Absent	No
Pointy	Round	Present	Yes

resultant tree →



different from before

Basic pseudocode :-

Given training set of size m
For b = 1 to B : ("B refers to no. of different trees you want")

Use sampling to create a new training set of size m. Train a decision tree on the new dataset.

B is usually set to a value between 64 - 128.

It never hurts to have a very large B but after some point you may end up with diminishing returns and make the algorithm slower.

- This particular instantiation of tree ensemble is called a bagged decision tree.

We might notice that even after sampling the features to split on at the root node and near the root node might not be the same.

To avoid this we add a modification to our algorithm. Suppose we have a set of N features, we introduce a randomness by forcing the algorithm to only choose from a random subset $k < n$ of those features at each split in each decision tree, thus improving the overall performance of the ensemble.

This algorithm gets the name "random forest algorithm".

A typical choice for value of k would be $k = \sqrt{n}$.

→ This algorithm works because bootstrap sampling allows us to explore all of the small changes happening to the algorithm and take its average.