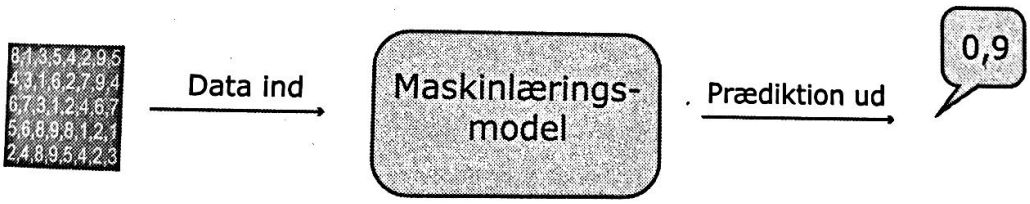
Det er udvikleren, der bestemmer, hvornår træningen er færdig, og det enkleste, hun kan gøre, er at sige, at træningen er slut, når modellen opnår en vis træfsikkerhed. En træfsikkerhed på 100 procent betyder, at modellen er i stand til at prædikere det rigtige svar på alle datapunkterne, mens 0 procent træfsikkerhed betyder, at modellen prædikerer forkert hver eneste gang. Tilfældige gæt, eller plat og krone med en mønt, giver en træfsikkerhed på 50 procent. Selvom en træfsikkerhed på 100 procent er næsten umulig at opnå, bør modellen alligevel kunne komme et godt stykke over 50 procent for at være nyttigere end en mønt. Hvor lang tid det tager, før modellen opnår en acceptabel træfsikkerhed, afhænger af, hvor vanskelig opgaven er at løse.

Når træningen er slut, sidder man tilbage med en maskinlæringsmodel, og alle verdens maskinlæringsmodeller fungerer ved at tage data ind, foretage beregninger og sende et resultat retur. Resultatet er altid et eller flere tal. Hvis vi siger, at "maskinen sagde nej", betyder det, at maskinen gav os et tal, som vi på forhånd har defineret som et "nej".

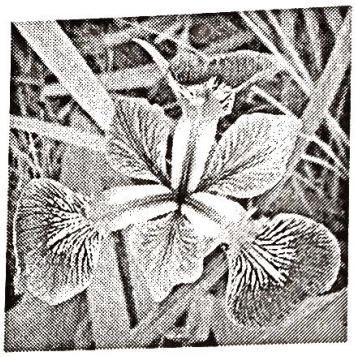


Blomster og beslutningstræer

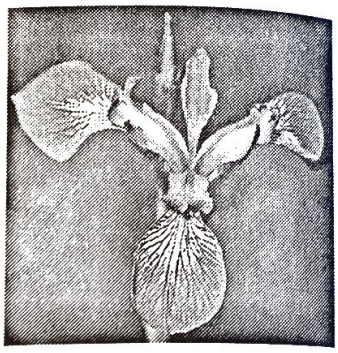
En stor del af de opgaver, vi gerne vil have maskiner til at løse for os, kan koges ned til det, vi kalder *klassificering* eller på jævnt sprog "se forskel på ting". Når din telefon afgør, om den ved synet af dit ansigt skal låse skærmen op eller ej, laver den i bund og grund en klassificering af det, den ser mellem de to kategorier "min ejers ansigt" og "ikke min ejers ansigt".

En enkel opgave at løse ved hjælp af maskinlæring – og som af en besynderlig grund er blevet en klassisk opgave for vordende dataloger – er at lave en model, som klassificerer tre blomster i irisfamilien, som til forveksling ligner hinanden. Desværre vokser ingen af dem vildt her i landet, men de kan købes i havecentre (jeg har tjekket), og du kan beundre dem på billederne herunder. De latinske navne er *Iris setosa*, *Iris versicolor* og *Iris virginica*.

Iris setosa



Iris versicolor



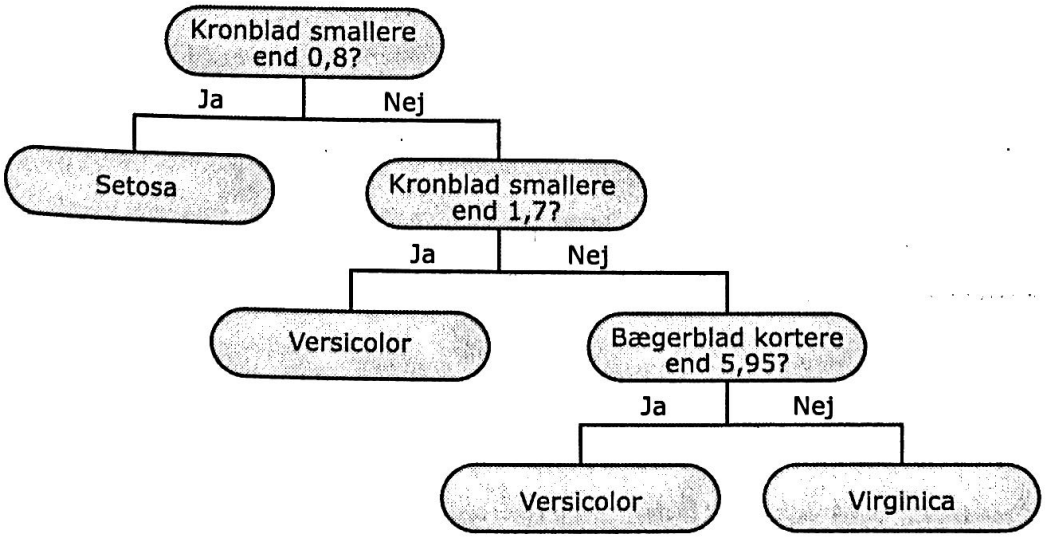
Iris virginica

Lad os forestille os, at jeg har en brændende lyst til at lave et computerprogram, som kan kende forskel på disse tre blomster. Havde vi befundet os i 1960, ville jeg have været nødt til at finde information om blomsternes forskellige egenskaber, lavet svarene om til regler og programmeret reglerne ind i computeren på samme måde som med banke-banke-på-vitserne. Takket være to ting er det meget nemmere i dag. For det første findes der data, som beskriver blomsterne: I 1936 foretog biologen Edgar Anderson observationer af mere end hundrede irisblomster og satte observationerne sammen med sine kategoriseringer af blomsterne ind i et datasæt. Dette datasæt er nu tilgængeligt på nettet, og alle, der har lyst, kan downloade det. Det indeholder 150 rækker, hvoraf de første seks ser sådan ud:

Længde bægerblad	Bredde bægerblad	Længde kronblad	Bredde kronblad	Art
4,6	3,2	1,4	0,2	Setosa
5,0	3,3	1,4	0,2	Setosa
7,0	3,2	4,7	1,4	Versicolor
6,4	3,2	4,5	1,5	Versicolor
7,6	3,0	6,6	2,1	Virginica
4,9	2,5	4,7	1,7	Virginica

Her har vi altså et datasæt, som indeholder information om tre forskellige irisblomster. Dette datasæt kan jeg give til en maskinlæringsalgoritme i min computer og bede om en maskinlæringsmodel, som er i stand til at kende forskel på de tre blomsterarter. Det eneste, jeg behøver at gøre, før maskinlæringen kan gå i gang, er at specificere, hvilken type maskinlæringsmodel jeg er ude efter. En absolut klassiker inden for maskinlæring er beslutningstræer. Det er digitale processkemaer, som opdeler data ved hjælp af ja/nej-spørgsmål, og ved hjælp af maskinlæring bestemmes svarene på ja/nej-spørgsmålene af maskinen baseret på data.

Jeg sætter gang i processen på min computer, og fordi der er tale om et enkelt problem, og datasættet er meget lille, tager træningen bogstaveligt talt under et minut, før min maskinlæringsmodel er klar. Baseret på vores data fra 1936 skabes følgende beslutningstræ:



Tro mig, jeg kan godt se, at det er underligt at skulle forholde sig til bredde og længde på bægerblade og kronblade i en bog om kunstig intelligens. Men jeg forsikrer, at det er et af de klassiske datasæt, som studerende i maskinlæring udvikler deres første beslutningstræer på, og det skyldes, at problemet er enkelt, men alligevel giver en god illustration af den måde, grundlæggende maskinlæring hyppigt anvendes på. Ofte bliver maskinlæringsmodeller udviklet af nogle, der ikke har ekspertviden om det, datasættet beskriver; de har brugt deres kræfter på at blive eksperter i maskinlæring. Selvom jeg ikke er i stand til at kende forskel på en løvetand og en blåklokke, kan jeg takket være tilgængelige data og en maskinlæringsalgoritme udvikle en maskinlæringsmodel, som gør mig i stand til at skelne mellem blomster, jeg ikke anede var forskellige. Bevæbnet med mit beslutningstræ og en lineal kan jeg nu gå ud i verden og se forskel på tre irisblomster, jeg egentlig ikke ved noget om. Men: Hvis modellen skal bruges til noget mere kritisk end hobbygartneri, er det derfor vigtigt, at jeg allierer mig med en ekspert i det, mine data beskriver, for at være sikker på, at de regler, beslutningstræet indeholder, giver mening.

Hvis du er den opmærksomme type, har du måske lagt mærke til to ting: Den ene er, at beslutningstræer i bund og grund bare er maskinskabte regelsystemer: Hver opdeling i træet, altså hvert spørgsmål, træet stiller til data, udgør en regel. Den andet er, at den nederste række i min tabel er forkert klassificeret: Blomsten er af typen virginica, men mit beslutningstræ siger, at det er en versicolor. Mit beslutningstræ vil komme til at lave flere af den slags fejl, fordi det ikke er stort nok til at foretage alle de nødvendige inddelinger. At lave et træ, som kategoriserer alle blomsterne helt rigtigt, ville kræve mange flere ja/nej-spørgsmål, og træet ville altså skulle have mange flere niveauer end det, jeg har lavet her.

Beslutningstræer har den store fordel, at de er intuitive for os mennesker. Vi kan kigge på dem og umiddelbart forstå, hvad de går ud på. Men hvis vi har et stort datasæt, eller der er komplicerede sammenhænge mellem data, kan et beslutningstræ have brug for

titusindvis af ja/nej-spørgsmål for at løse sin opgave og således være alt andet end enkelt at forstå for mennesker. Alligevel forstår vi, at beslutningstræer bruger symboler, som giver mening for os mennesker, og at der derfor er tale om en symbolsk metode. Men: Der findes mange typer opgaver, som beslutningstræer aldrig vil kunne løse, som for eksempel at se forskel på billeder eller beherske menneskesprog. Og hvis beslutningstræerne befinder sig i den ene ende af skalaen, når det handler om at være intuitivt forståelige for mennesker, skal vi helt over i den anden ende af skalaen for at finde de maskinlæringsmodeller, som lærer sig at løse det, der har vist sig at være de vanskeligste problemer inden for kunstig intelligens. Vi skal se på en subsymbolsk metode til maskinlæring, som har vist sig at have en helt unik evne til at løse problemer, nemlig neurale netværk.

Subsymbolsk AI

Maskinlæringsmodeller findes i mange forskellige varianter, men i løbet af de sidste 20 år har man fået øjnene op for neurale netværk. Når dagens maskiner genkender ansigter, laver kunst, skriver tekst, drømmer eller laver falske data, er det takket være neurale netværk.

Som alle maskinlæringsmodeller får neurale netværk data ind, laver deres beregninger og leverer en prædiktion. Det, der gør neurale netværk til noget særligt, er måden, de laver beregninger på. Neurale netværk opbygges ved at sammensætte små beregningsenheder, som vi kalder *noder*. En node gør tre ting: Den tager imod et tal, udfører en matematisk operation på tallet, og så sender den resultatet af operationen ud. En sådan operation kan være "tjek, om tallet er mindre end 0. Hvis ja, send 0 ud. Hvis tallet er større end 0, send det tal ud, som kom ind". Det kan lyde frygteligt banalt, men ved at sammensætte mange af den slags noder til et netværk kan vi lave verdens mest kraftfulde maskinlæringsmodeller. Den enkleste måde at sammensætte noder på er ved at stable dem oven på hinanden sådan her: