

# Föreläsning 3 - Trädmodeller

Josef Wilzen

2021-08-30

# Outline

- 1 Introduktion
- 2 Trädmodeller
- 3 Metoder för beslutsträd
- 4 Regularisering
- 5 Kommentarer om beslutsträd

# Introduktion

- Denna vecka:
  - ▶ Trädmodeller
  - ▶ Naive Bayes, k närmaste grannar, ensemblemetoder
- Nästa vecka: Neurala nätverk

# Övervakad inlärning - Klassificering

- Målet är att dela upp objekt i ett antal förutbestämda klasser
- Ex:
  - ▶ Upptäcka spam-mail baserat på texten i mailen
  - ▶ Klassificera god- eller elakartad tumör baserat på medicinska bilder

# Exempeldata

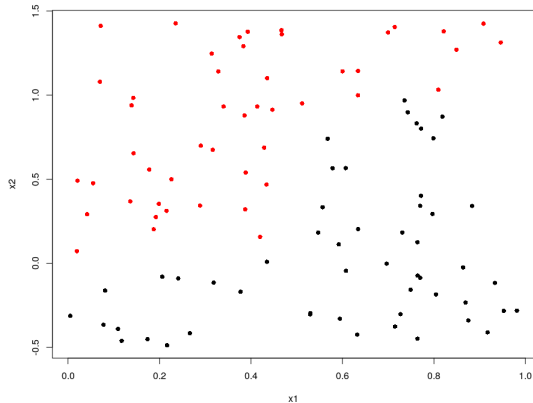
Klassificerarens uppgift är att anpassa en målfunktion  $f$  (att bygga upp en modell  $f$ ) som kartlägger varje attributmängd  $x$  till en av förbestämda klasser  $y$

| ID         | x1               | x2         | x3          | x4               | x5              | x6       | x7         | y           |
|------------|------------------|------------|-------------|------------------|-----------------|----------|------------|-------------|
| Name       | Body temperature | Skin cover | Gives birth | Aquatic creature | Aerial creature | Has legs | Hibernates | Class label |
| human      | warm-blooded     | hair       | yes         | no               | no              | yes      | no         | mammal      |
| python     | cold-blooded     | scales     | no          | no               | no              | no       | yes        | non-mammal  |
| salmon     | cold-blooded     | scales     | no          | yes              | no              | no       | no         | non-mammal  |
| whale      | warm-blooded     | hair       | yes         | yes              | no              | no       | no         | mammal      |
| frog       | cold-blooded     | none       | no          | semi             | no              | yes      | yes        | non-mammal  |
| komodo     | cold-blooded     | scales     | no          | no               | no              | yes      | no         | non-mammal  |
| bat        | warm-blooded     | hair       | yes         | no               | yes             | yes      | yes        | mammal      |
| pigeon     | warm-blooded     | feathers   | no          | no               | yes             | yes      | no         | non-mammal  |
| cat        | warm-blooded     | fur        | yes         | no               | no              | yes      | no         | mammal      |
| shark      | cold-blooded     | scales     | yes         | yes              | no              | no       | no         | non-mammal  |
| turtle     | cold-blooded     | scales     | no          | semi             | no              | yes      | no         | non-mammal  |
| penguin    | warm-blooded     | feathers   | no          | semi             | no              | yes      | no         | non-mammal  |
| porcupine  | warm-blooded     | quills     | yes         | no               | no              | yes      | yes        | mammal      |
| eel        | cold-blooded     | scales     | no          | yes              | no              | no       | no         | non-mammal  |
| salamander | cold-blooded     | none       | no          | semi             | no              | yes      | yes        | non-mammal  |

# Linjära och Icke-linjära modeller

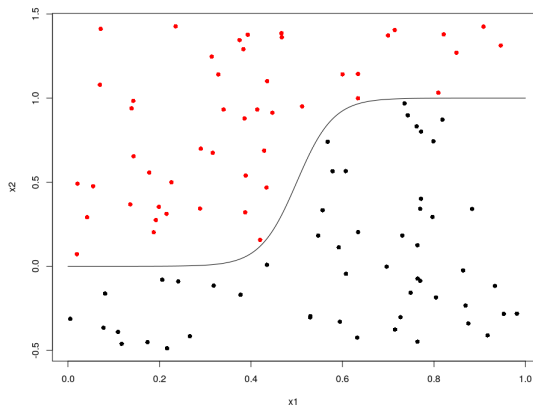
- Klassificering och regression:
- Linjära modeller:
  - ▶ Linjär regression
  - ▶ (Linjär) logistisk regression
  - ▶ Lätta att skatta och tolka
  - ▶ Vissa problem går inte att lösa!

# Linjära och Icke-linjära modeller



100 obs, två förklarande variabler  $x_1$  och  $x_2$ ,  $y$  är binär

# Linjära och Icke-linjära modeller



Beslutsregel:

$$x_2 \cdot (1 + \exp(-25 \cdot (x_1 - 0.5))) > 1$$



# Linjära och Icke-linjära modeller

Linjär regression:

- givet  $X = (x_1, x_2, \dots, x_p)$ ,  $y$ :  $y = X\beta$
- Vi kan transformera variablerna i  $X$
- Polynomregression:  $X = (x, x^2, x^3, \dots, x^p)$
- Andra exempel:  $\log(x)$ ,  $\sqrt{x}$ ,  $\cos(x)$ ,  $\exp(x)$ , interaktioner, stegfunktioner, diskretisering, dummy-kodning
- Kallas i maskininlärning för “feature engineering”
  - ▶ Svårt att veta transformation vi ska göra för ett givet problem!
  - ▶ Svårt med komplexa datastrukturer: text, bilder mm

# Linjära och Icke-linjära modeller

Inom maskininlärning:

- Olika metoder för att kunna anpassa mer generella icke-linjära modeller
- Vi vill hitta “automatiska” transformationer av de förklarande variablerna
  - ▶ Som hjälper oss att prediktera  $y$
- Vi vill kunna hantera många variabler, och av olika typ.

# Icke-linjära modeller

Exempel:

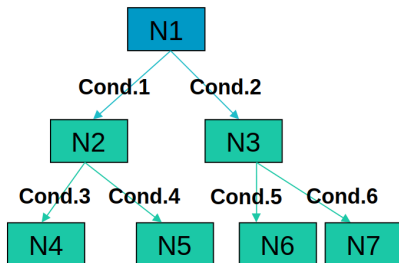
- Trädmodeller
- Neurala nätverk
- Splines
- Local regression
- Generalized additive models (GAM)
- Support vector machines
- K-närmaste grannar

# Trädmodeller

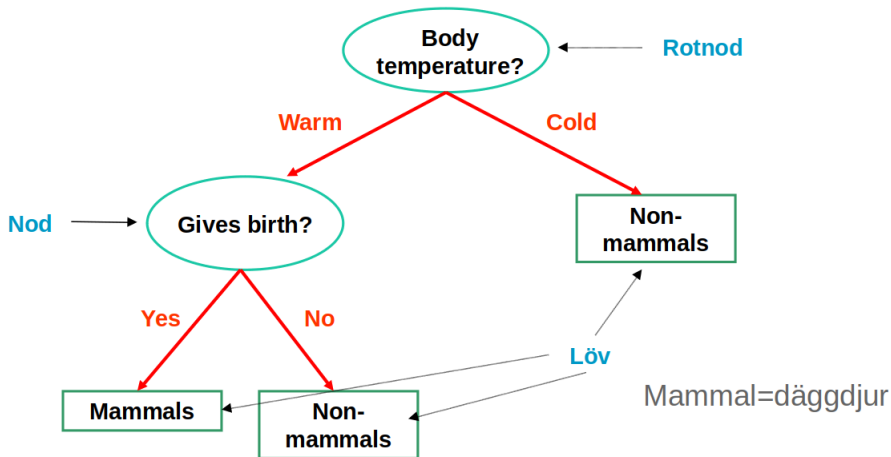
- Modellen defineras som ett träd (datorstruktur), ofta binärt
- Icke-parametrisk metod
- Två steg:
  - ① Dela upp variabelrummet i icke överlappade regioner  $\{R_1, R_2, R_3, \dots, R_J\}$ : axelparallella rektanglar
  - ② Alla obs i en region har samma anpassade värde
- Både regression och klassificering
- Hur ska vi dela upp? Viktiga principer:
  - ▶ recursive binary splitting
  - ▶ top-down, greedy

# Beslutsträdets uppbyggnad

- Rotnod (N1)
- Noder (alla N)
- Löv/slutnoder (N4-N7)
- Regler
- Varje löv har ett tilldelat klassvärde baserat på någon röstningsmetod



# Exempel

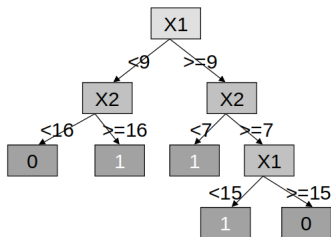
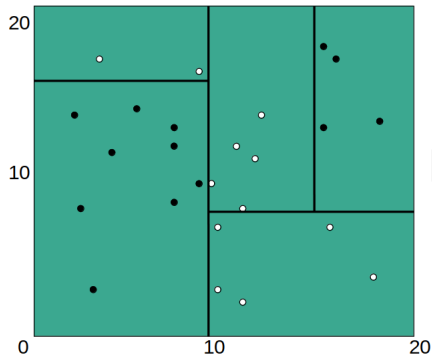


# Hur bygger man upp ett beslutsträd?

## Hunt's algorithm

- Givet Datamängd  $D_t = \{(X_{1i}, \dots, X_{pi}, Y_i), i = 1 \dots n\}$ ,  $t$ -akuell nod
- Om alla  $Y_i$  är lika, markera  $t$  som löv med klassvärde  $Y_i$
- Om inte, använd **testregeln** för att dela upp  $D_t$  i  $D_{t1} \dots D_{tn}$ , och sedan kör  $\text{Hunt}(D_{t1}, t_1), \dots, \text{Hunt}(D_{tn}, t_n)$  tills alla noder fått en klass
  - ▶ Recursive partitioning

# Exempel





# Hunt's algoritm forts.

- Alternativt avslutskriterium:
  - ▶ Alla objekt i en nod har identiska attributvärden
  - ▶ Noden deklarerar som ett löv med klassvärde av majoriteten
- Olika testregler:
  - ▶ Binära attribut → binär uppdelning
  - ▶ Nominala attribut → binär eller mångfaldig uppdelning
  - ▶ Ordinala attribut → uppdelning som bevarar attributsföljden
  - ▶ Interval-attribut → uppdelning till uteslutande intervall

# Sammanfattning av skapandet

- Dela upp observationer för att separera angivna klasser
  - ▶ Olika testregler ska jämföras
- Att avsluta processen
  - 1 Fortsätt dela upp i noder tills alla observationer i löven hör till samma klass, eller har exakt samma attributvärden
  - 2 Bestäm en regel för tidig avslutning

# CART

- CART = Classification and Regression Tree
- Skillnader mellan kontinuerliga och diskreta utfall

# CART - Regressionsträd

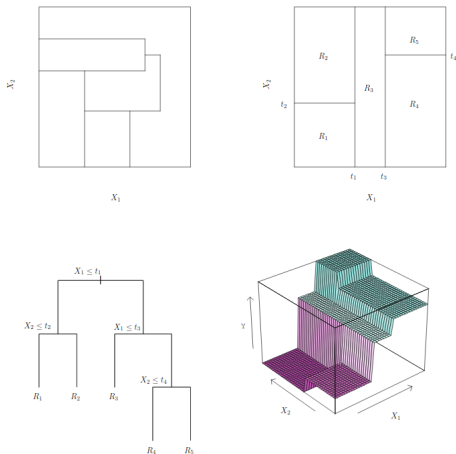
- Minimera felfunktionen  $\sum (y_i - f_i)^2$  är dyrt för alla olika  $f_i$  och uppdelningar
- Istället: Leta efter en variabel ( $x$ ) och en uppdelning ( $s$ ) som minimerer

$$R_1(j, s) = \{x | x_j \leq s\} \quad R_2(j, s) = \{x | x_j > s\}$$

$$\min_{j,s} \left[ \min_{c_1} \sum_{x_i \in R_1(j,s)} (y_i - c_1)^2 + \min_{c_2} \sum_{x_i \in R_2(j,s)} (y_i - c_2)^2 \right]$$

$c_j$  skattas ofta som medelvärdet av obs i region  $R_j$ .

# CART - Regressionsträd



**FIGURE 8.3.** Top Left: A partition of two-dimensional feature space that could not result from recursive binary splitting. Top Right: The output of recursive binary splitting on a two-dimensional example. Bottom Left: A tree corresponding to the partition in the top right panel. Bottom Right: A perspective plot of the prediction surface corresponding to that tree.

# CART - Klassificeringsträd

Vi behöver ett mått för att utvärdera om en regel är bra eller inte!

Definiera proportioner:

$$p_k = \frac{1}{s} \sum_{i=1}^s 1_{(y_i=k)}$$

Definiera föroreningsmått:

- Entropy =  $\sum_{i=0}^{c-1} p_i \cdot \log_2(p_i)$ 
  - ▶ Små värden bra
- Gini =  $1 - \sum_{i=0}^{c-1} p_i^2$ 
  - ▶ Total varians för alla klasser, små värden bra
- Felkvot (misclassification error) =  $1 - \max(p_i)$ 
  - ▶ Små värden bra

# CART - Klassificeringsträd

Välj uppdelning som maximerar informationsvinst

$$\Delta = I(\text{parent}) - \sum_{j=1}^k \frac{N(v_j)}{N} I(v_j)$$

- där  $I(\cdot)$  är ett föroreningsmått
- $N$  är antal objekt i föräldranoden
- $v_j$  är barnnod  $j$
- $\frac{N(v_j)}{N}$  relativa vikter för varje barnnod
- $N(v_j)$  är antal objekt

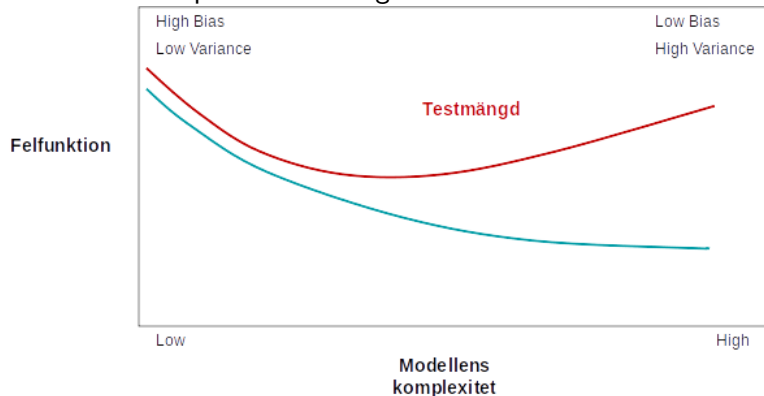
# Klassificeringsträd

- Prediktioner görs med majoritetsröstning om det finns blandade klasser i ett löv
- Vi vill att löven i så hög utsträckning ska ha obs med en klass
- Entropy och Gini är bättre föroeringsmått, då ge ger renare löv
  - ▶ Används vid träning
- Felkvot använd ofta för att utvärdera modellen på testdata



# Regularisering

Trädmodeller överpassar lätt! Hög varians!



# Hur motverkas överanpassning?

- Förbeskärning (pre-pruning):
  - ▶ Sluta expandera trädet när informationsvinsten är lägre än en vald tröskel
  - ▶ Kräv ett visst minsta antal obs i varje löv
  - ▶ Problem: Vilken tröskel ska väljas? Hantera liten vinst ett steg → stor vinst nästa steg?
- Efterbeskärning (post-pruning):
  - ▶ Beskär ett helt utväxt träd → ersätt ett delträd med ett löv

## Efterbeskrning - Klassificeringsträd

Ett stort träd  $\rightarrow$  komplex modell  $\rightarrow$  överanpassning

- 1 Välj ett delträd,  $T$ , där  $|T|$  är antalet löv i  $T$
- 2 Minimera

$$C_{\alpha}(T) = \sum_{v \in \text{Löv i } T} N(v) \cdot I(v) + \alpha |T|$$

- Detta kallas för “cost complexity pruning”
- Använd korsvalidering för att skatta  $\alpha$ , välj det värde som ger minst valideringsfel
- Notera: detta är likt idén i lasso

# Skatta regressionsträd med cost complexity pruning

---

**Algorithm 8.1** *Building a Regression Tree*

---

1. Use recursive binary splitting to grow a large tree on the training data, stopping only when each terminal node has fewer than some minimum number of observations.
  2. Apply cost complexity pruning to the large tree in order to obtain a sequence of best subtrees, as a function of  $\alpha$ .
  3. Use K-fold cross-validation to choose  $\alpha$ . That is, divide the training observations into  $K$  folds. For each  $k = 1, \dots, K$ :
    - (a) Repeat Steps 1 and 2 on all but the  $k$ th fold of the training data.
    - (b) Evaluate the mean squared prediction error on the data in the left-out  $k$ th fold, as a function of  $\alpha$ .Average the results for each value of  $\alpha$ , and pick  $\alpha$  to minimize the average error.
  4. Return the subtree from Step 2 that corresponds to the chosen value of  $\alpha$ .
-

# Kommentarer om beslutsträd

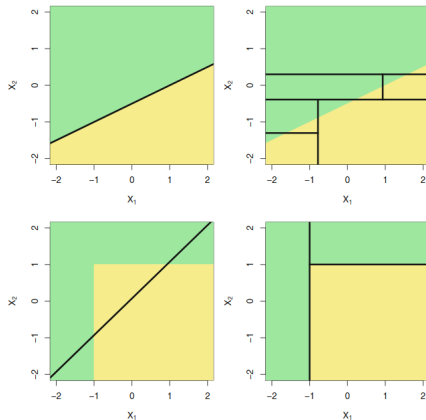
Linjär regression:

$$f(x) = \beta_0 + \sum_{i=1}^p x_i \beta_i$$

Regressionsträd:

$$f(x) = \sum_{i=1}^M c_i 1_{(x \in R_i)}$$

# Kommentarer om beslutsträd



**FIGURE 8.7.** Top Row: A two-dimensional classification example in which the true decision boundary is linear, and is indicated by the shaded regions. A classical approach that assumes a linear boundary (left) will outperform a decision tree that performs splits parallel to the axes (right). Bottom Row: Here the true decision boundary is non-linear. Here a linear model is unable to capture the true decision boundary (left), whereas a decision tree is successful (right).

# Kommentarer om beslutsträd

## Fördelar:

- Lätta att förstå och tolka
- Klarar av olika responsvariabler
- Kräver inte så mycket datahantering innan
- Funkar på relativt stora dataset
- Icke-parameterisk metod: antal “parametrar” beror på data
- Automatisk variabelselektion
- Kan anpassa många olika sorters funktioner
  - ▶ Klarar av olika sorters variabler
  - ▶ Starkt korrelerade attribut påverkar inte

# Kommentarer om beslutsträd

Nackdelar:

- Sämre prediktiv förmåga än vissa andra metoder
- Orubusta: överanpassar lätt
- Omöjligt att hitta det optimala trädet pga snåla algoritmer
- Vissa enkla funktioner kräver komplext träd: tex en linjär funktion



# Kommentarer om beslutsträd

Förbättringar:

- Bagging
- Random forest
- Boosting
- BART: Bayesian Additive Regression Trees
  - ▶ Inte denna kurs

# Avslut

- Frågor? Kommentarer?
- Kurshemsidan
- Labben