LINKÖPING UNIVERSITY

732G12 Data Mining

Föreläsning 3

Johan Alenlöv
IDA, Linköping University, Sweden

Dagens föreläsning

- Linjära och icke-linjära modeller
- Trädmodeller
- Metoder för besultsträd
- Regularisering

Linjära och icke-linjära modeller

Inom både regression och klassificering har vi pratat om linjära modeller.

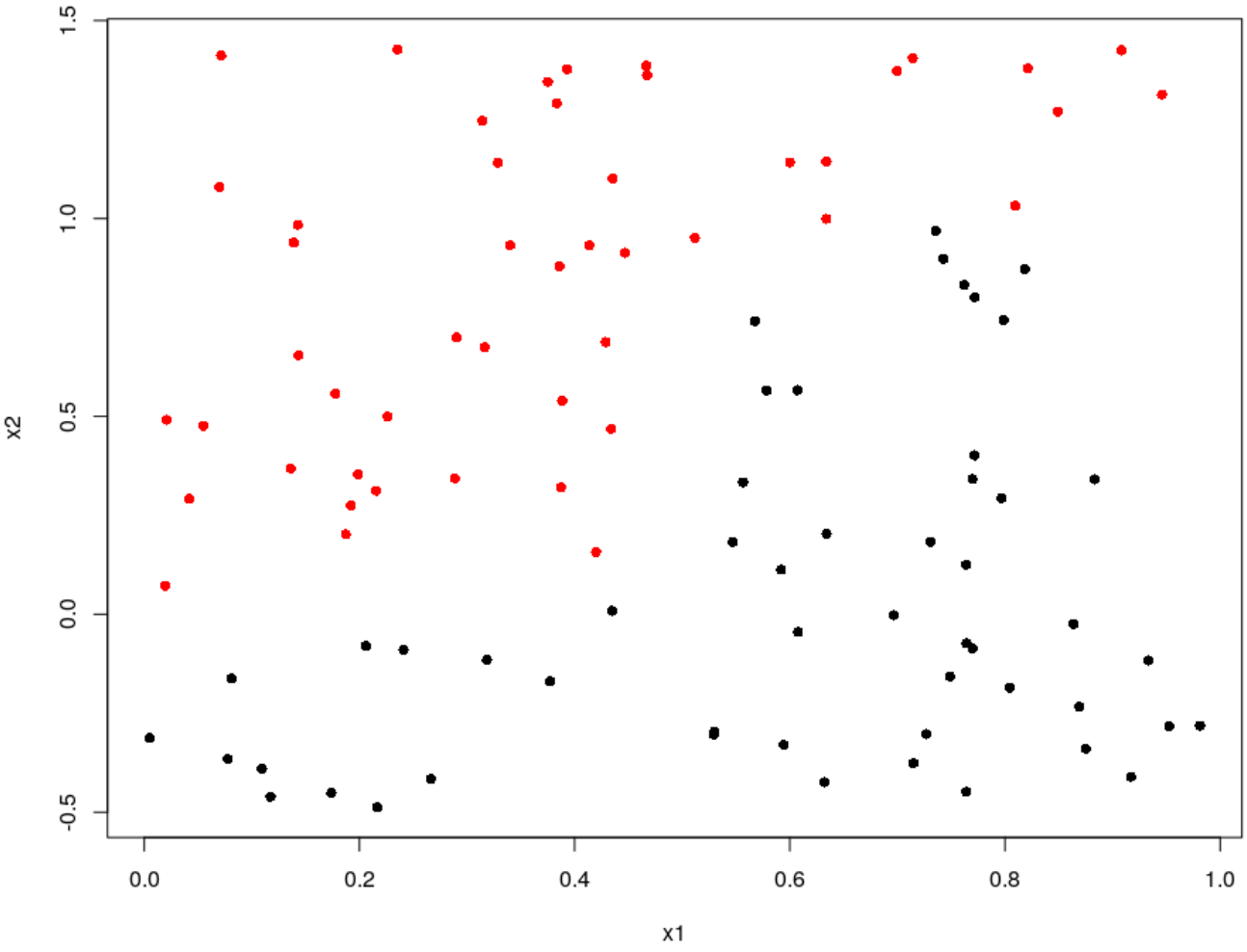
Exempel:

- Linjär regression
- Linjär logistisk regression

Lätta att skatta och tolka.

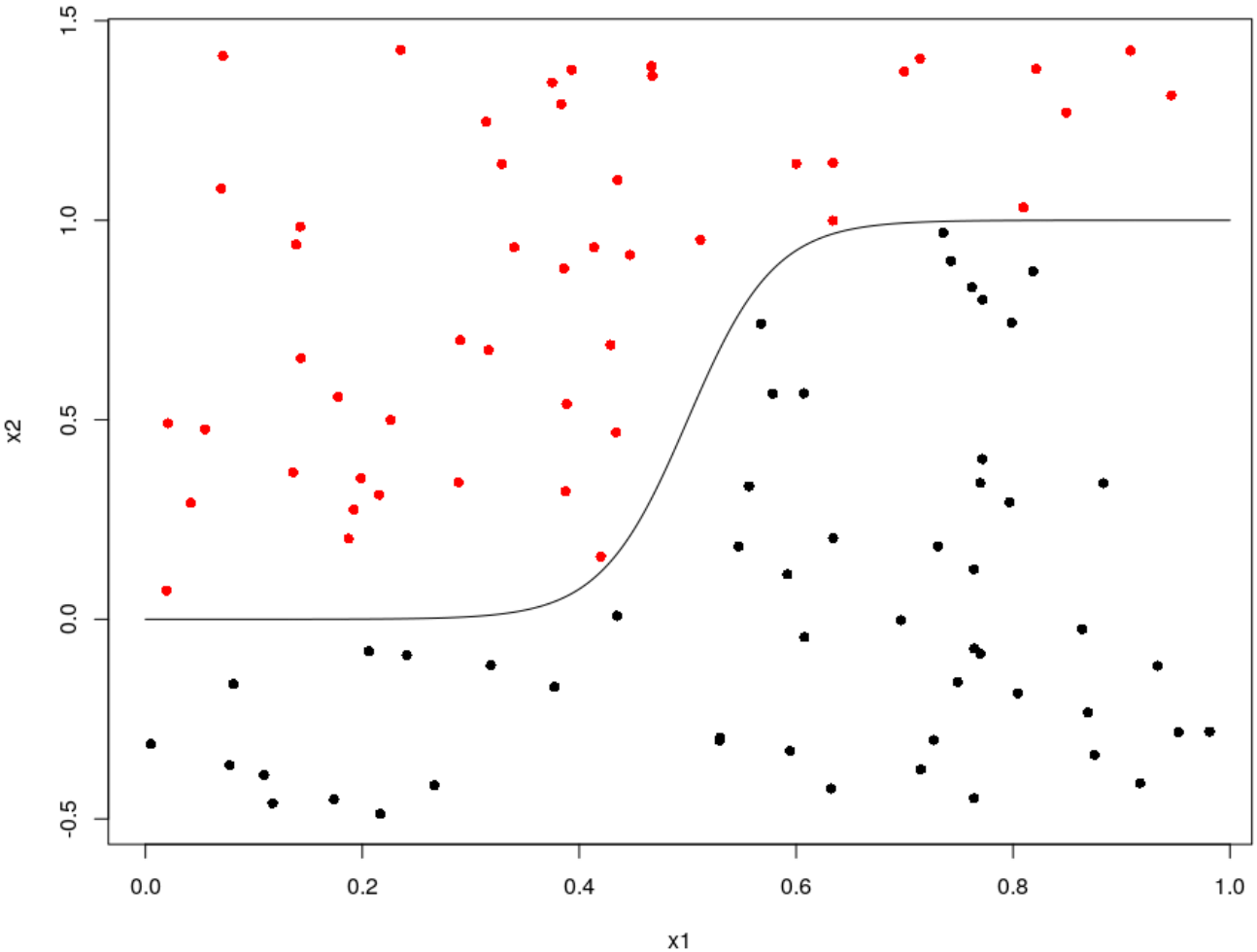
Kan inte lösa alla problem.

Exempel - Klassificering



100 observationer, två förklarande variabler, binär respons.

Exempel - Klassificering



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

Bästa linjära?

Linjär regression till icke-linjär?

Som vanligt, har förklarande variabler $\mathbf{X} = (x_1, x_2, \ldots, x_p)$ och respons y.

Modellerear som $y = \mathbf{X}\beta$.

Ett sätt att hantera icke-linjära samband är att transformera $\mathbb{X}$.

Till exempel:

- Polynomregression
- Andra funktioner
- Interaktioner
- Stegfunktioner
- Diskretisering

Svårt att veta vilka transformationer man ska göra, svårt att transformera komplexa datastrukturer.

Icke-linjära modeller

Maskininlärning har gett oss många olika metoder för att anpassa mer generella icke-linjära modeller.

Målet är att hitta ”automatiska” transformationer av de förklarande variablerna.

Ska kunna hantera många variabler av olika typer.

Exempel:

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

Trädmodeller

Idé: Dela upp variabelrummet i icke överlappande regioner (rektanglar), alla observationer i samma region har samma värde.

Reglerna för att hitta sin region kan beskrivas i en trädstruktur (binärt träd) kallas det för trädmodeller.

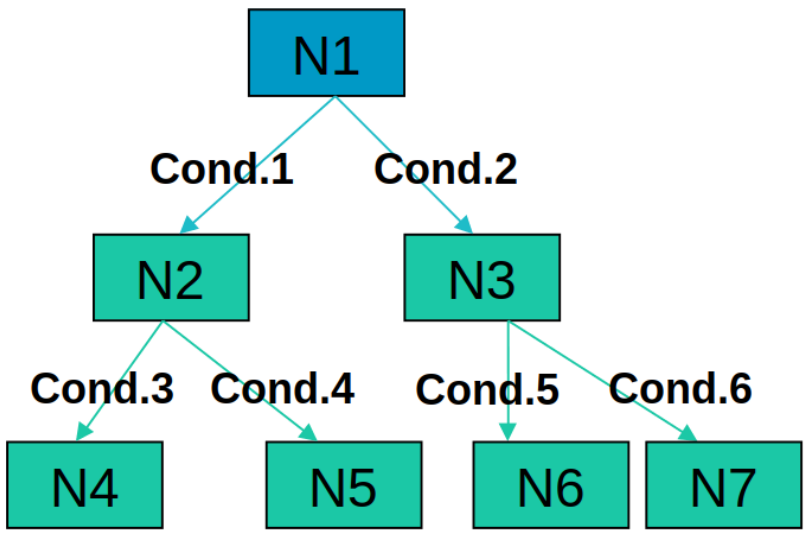
Skattning inom ett område sker oftast genom medelvärde eller typvärde.

Hur delar vi upp variabelrummet på ett bra sätt?

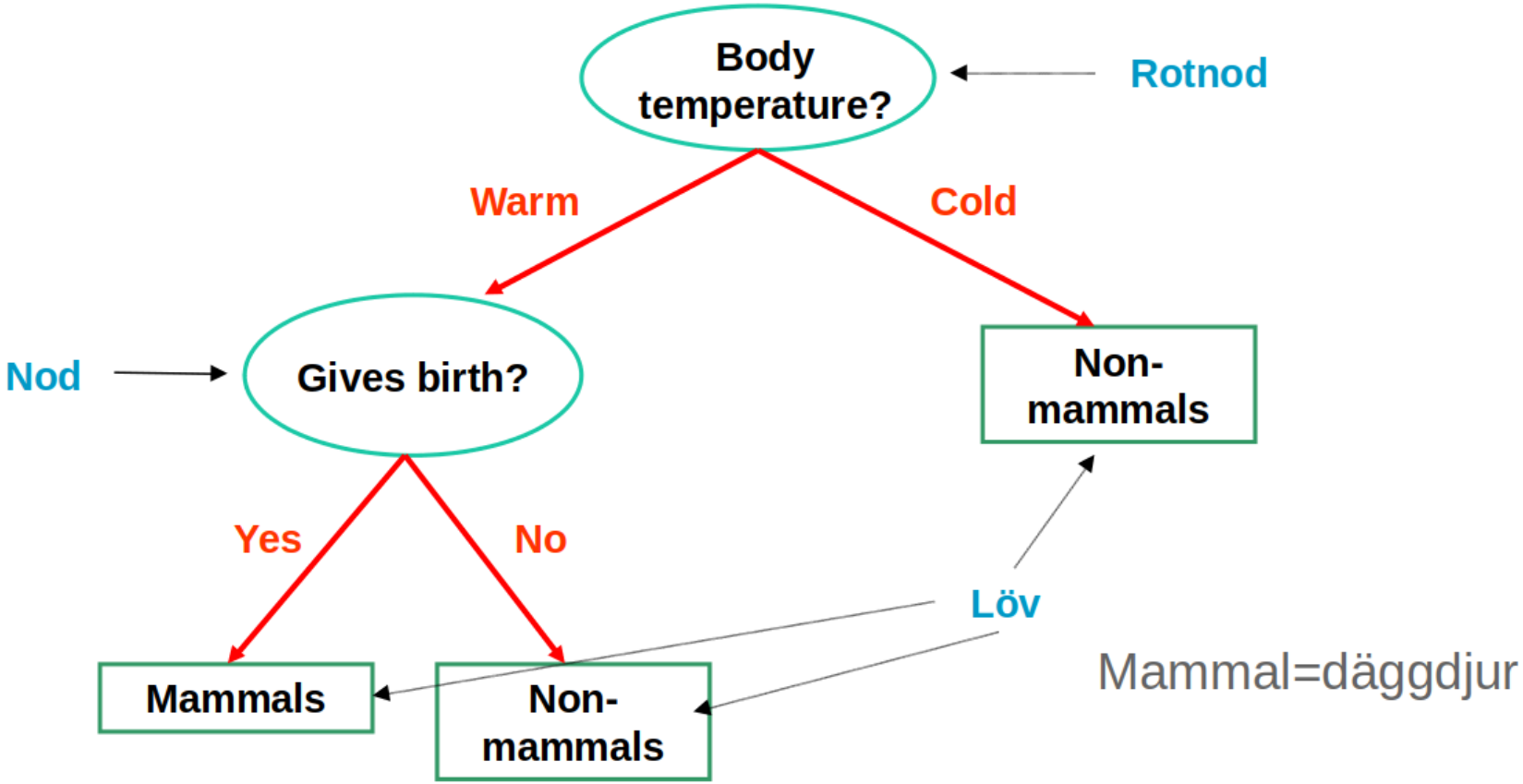
Besultsträd

Ett träd består av:

- Rotnod (N1)
- Noder (N*)
- Löv/slutnoder (N4-N7)
- Regler (Cond.1-Cond.6)
- Varje löv har ett tilldelat klassvärde



Beslutsträd Exempel



Att bygga ett träd

Hunt's algoritm

1. Givet en nuvarande datamängd $D_t = \{(X_i, Y_i), i = 1, \ldots, n\}$ för den aktuella noden t.
2. Om alla Y_i är lika, markera t som ett löv och ge värdet Y_i.
3. Annars, använd en **testregel** för att dela upp D_t i flera delar $D_{t_1}, \ldots, D_{t_k}$ och kör algoritmen (steg 1) för alla dessa noder.

Testregler:

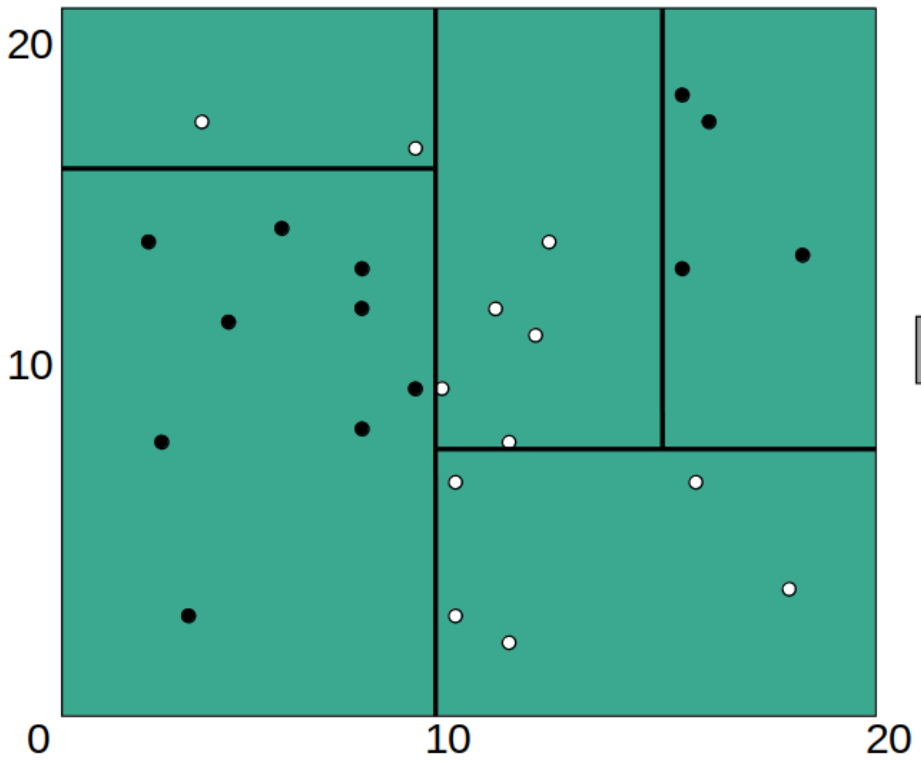
Binära attribut Binär uppdelning

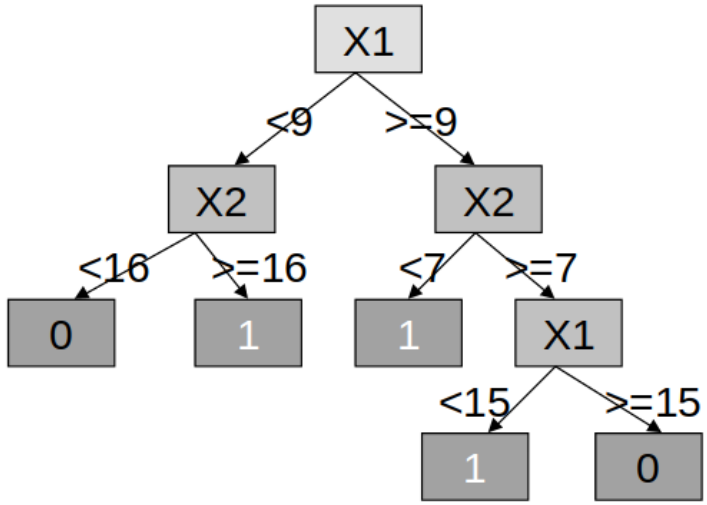
Nomiala attribut Binär eller mångfaldig uppdelninig

Ordinala attribut Uppdelning som bevarar attributsföljden

Intervall attribut Uppdelning till icke-överlappande intervall

Att bygga ett träd - Exempel





Trädmodeller

Sammanfattninig:

- Idé: Dela upp observationerna för att separera klasser.
- Uppdelningen sker genom att jämföra testregler.
- För att avsluta processen:
  - Dela upp tills alla observationer har samma klass
  - Alt 1. Dela upp tills alla attribut är lika.
  - Alt 2. Bestäm en regel för tidigt avslut.

CART

Classification and Regression Trees (CART).

I grunden Hunt's algoritm.

Stöd för kontinuerliga och diskreta utfall.

Optimering för att välja bästa splitten.

CART - Regressionsträd

Att försöka minimera RSS över alla möjliga partitioneringar och funktionsvärden är orimligt.

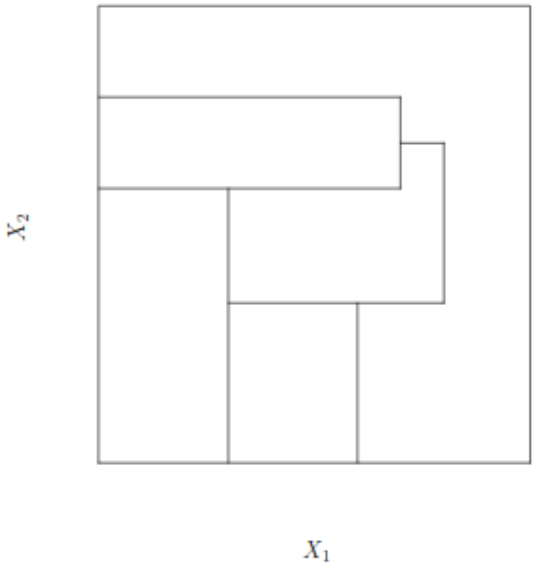
Istället kör vi med en girig algoritm, där vi vill hitta bästa variabeln x_j och uppdelning s genom att lösa följande problem:

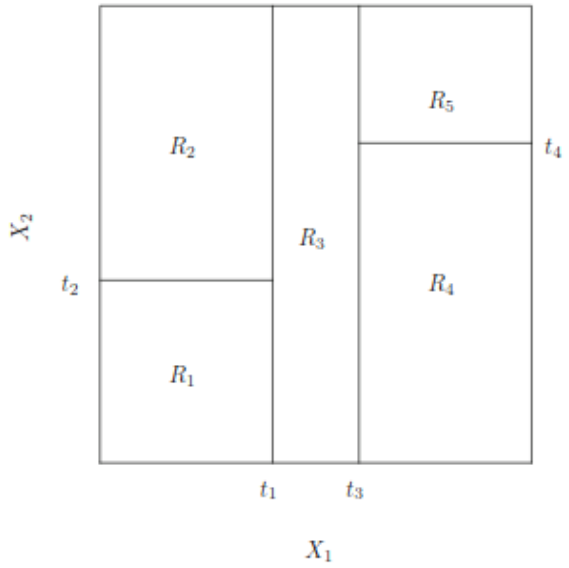
$$\min_{j,s} \left[\min_{c_1} \sum_{x_1 \in R_1(j,s)} (y_i - c_1)^2 + \min_{c_2} \sum_{x_i \in R_2(j,s)} (y_i - c_2)^2 \right],$$

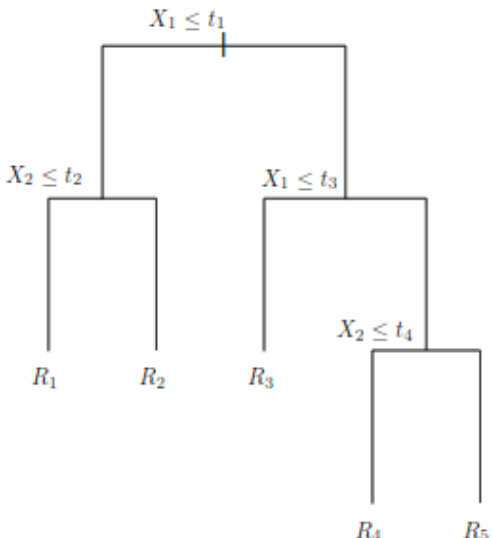
$$R_1(j,s) = \{x \,|\, x_j \leq s\} \qquad R_2(j,s) = \{x | x_j \geq s\},$$

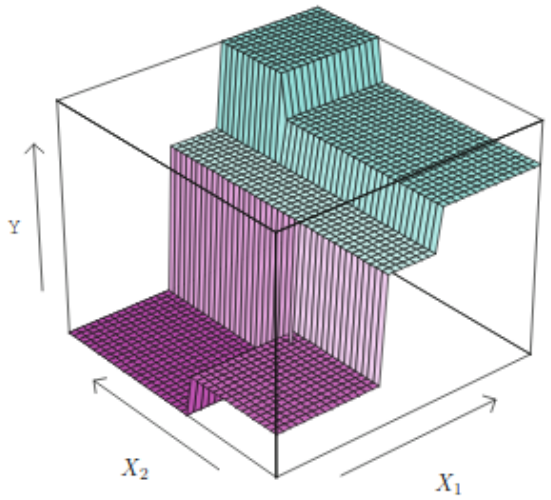
c_1 och c_2 skattas oftast som medelvärdet av respektive region.

CART - Regressionsträd









CART - Klassificeringsträd

När det kommer till klassificering behöver vi ett nytt mått för att utvärdera om en regel är bra eller dålig.

Låt $p_{m,k} = \frac{1}{s} \sum_{i:x_i \in R_m} \mathbb{I}_{y_i = k}$ vara proportionen av träningsdata i region m som tillhör klass k.

Kriterier för att välja regioner:

Felkvot $E = 1 - \max_k(p_{m,k})$

Gini index $G = \sum_k p_{m,k}(1 - p_{m,k}) = 1 - \sum_k p_{m,k}^2$

Cross-entropy $D = -\sum_k p_{m,k} \log(p_{m,k})$

CART - Klassificeringsträd

Välj nu den uppdelning som maximerar informationsvinsten

$$\Delta = I(\text{förälder}) - I(\text{barn}),$$

$$\Delta = I(\text{förälder}) - \sum_j \frac{N(R_j)}{N} I(R_j),$$

Där

$I(\cdot)$	är ditt valda mått (Felkvot, Gini, Entropy)
N	antal objekt i föräldranoden
R_j	är barnnod j
$N(R_j)$	antal objekt i barnnod j

CART - Klassificeringsträd

Predikationer görs med majoritetsröstninig. Vi vill helst att alla observationer i ett löv ska ha samma klass.

Vi stoppar utbyggnadet på samma sätt som i Hunt's algoritm.

Entropy och Gini är bättre mått, de ger renare löv.

Felkvoten används ofta för att utvärdera trädet på testdata.

Trädmodeller överanpassar väldigt lätt. Vilket ger en hög varians!

Motverka överanpassning

Två olika sätt

Förbeskärnining
- Sluta expandera trädet när informationsvinsten är lägre än en vald tröskel.
- Kräv ett visst minsta antal obs i varje löv.

Efterbeskärninig
- Beskär ett helt utväxt träd, ersätt delträd med ett löv.

Efterbeskärning

Använd CART för att ta växa fram ett stort träd T_0 på all träningsdata.

För varje $\alpha \geq 0$ finns ett delträd $T \subset T_0$ som minimerar kostande

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

där $|T|$ är antalet löv i T.

Använd korsvalidering för att skatta α, välj det som ger minst valideringsfel.

Idén är lik LASSO.

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 α.

3. Use K-fold cross-validation to choose α. That is, divide the training observations into K folds. For each $k = 1, \ldots, K$:

   (a) Repeat Steps 1 and 2 on all but the kth fold of the training data.

   (b) Evaluate the mean squared prediction error on the data in the left-out kth fold, as a function of α.

   Average the results for each value of α, and pick α to minimize the average error.

4. Return the subtree from Step 2 that corresponds to the chosen value of α.

Trädmodell eller linjär modell

För regression har vi:

Linjär regression

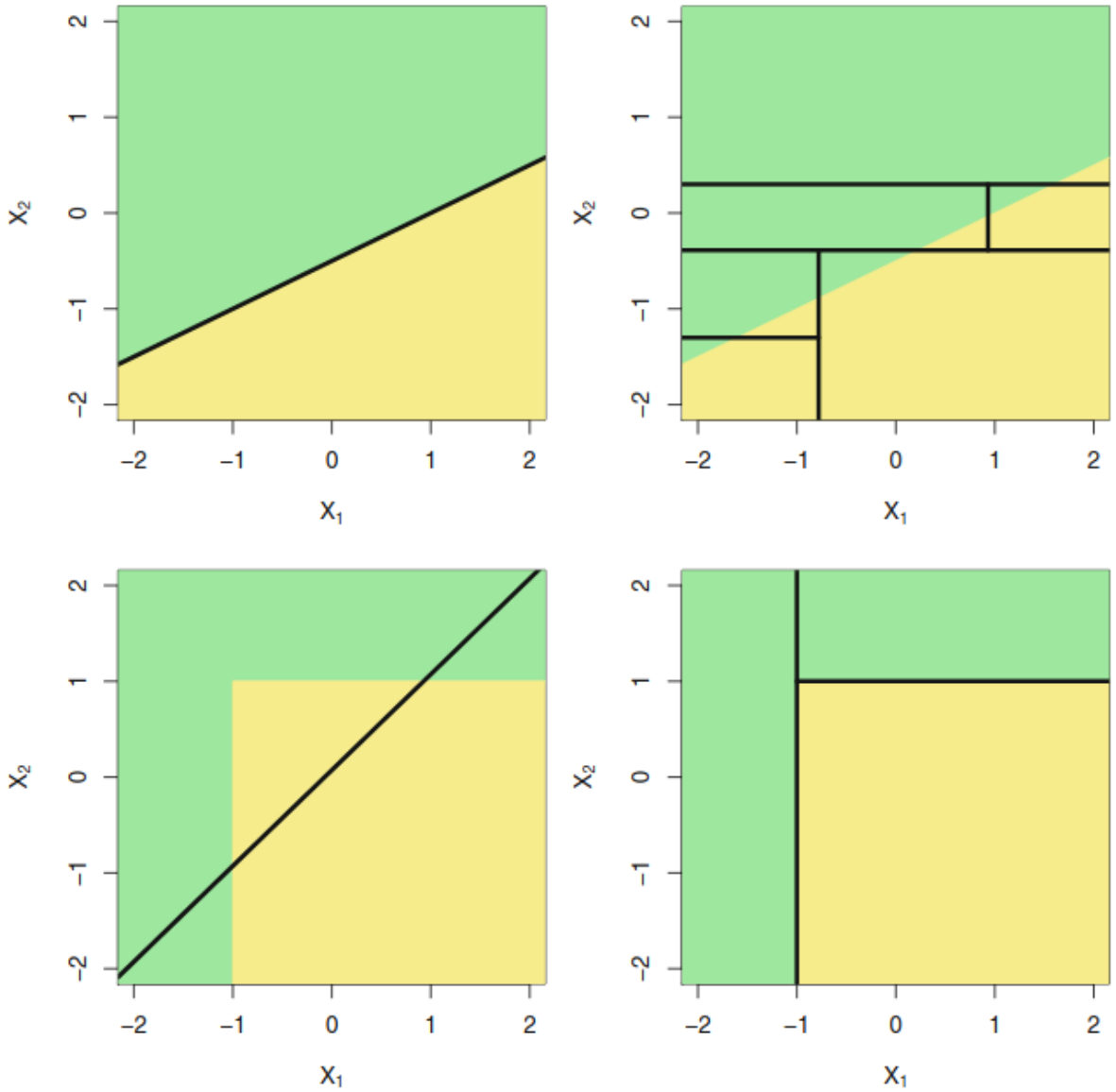
$$f(x) = \beta_0 + \sum_{i=1}^{p} x_i \beta_i$$

Regressionsträd

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

Trädmodell eller linjär modell

För klassificering har vi:



Kommentarer om trädmodeller

Fördelar:

- Lätta att förstå och tolka
- Klarar av olika responsvariabler
- Kräver inte så mycket datahantering innan
- Funkar på relativt stora dataset
- Icke-parametrisk metod
- Automatisk variabelselektion
- Kan anpassa många olika sorters funktioner

Nackdelar:

- Sämre prediktiv förmåga än vissa andra metoder
- Orubusta: överanpassar lätt
- Omöjligt att hitta det optimala trädet
- Vissa enkla funktioner kräver ett komplext träd

Kommentarer om trädmodeller

Förbättringar:

- Bagging
- Random forest
- Boosting
- BART: Bayesian Additive Regression Trees