Political Data Science

Lektion 11: Superviseret læring II

Undervist af Jesper Svejgaard, foråret 2018
Institut for Statskundskab, Københavns Universitet
github.com/jespersvejgaard/PDS

I dag

1. Overblik og opsamling fra sidst
2. Dagens pensum
3. Workshop
4. Afrunding og næste gang

Overblik

1. Intro til kurset og R
2. R Workshop I: Explore
3. R Workshop II: Import, tidy, transform
4. R Workshop III: Programmering & Git
5. Web scraping & API
6. Tekst som data
7. Visualisering
8. GIS & spatiale data
9. Estimation & prædiktion
10. Superviseret læring I
11. Superviseret læring II
12. Usuperviseret læring
13. Refleksioner om data science
14. Opsamling og eksamen

Fokus i dag

Konceptuelt:

- Klassifikations- og regressionstæer
- Ensemble-metoder
- `caret`

Workshop:

- Challenge: Byg en bedre prædiktionsmodel end din underviser!

Opsamling fra sidst

Opsamling fra sidst

Øvelser

Find løsning på opgaverne fra sidste uge på Github:

- `PDS/scripts/10_script.R`

Opsamling fra sidst

Undervisning

Hovedpointer:

- Logistisk regression
- Regularisering
- Tuning
- Resampling-metoder

(… hvad I altsammen får brug for i challengen!)

Regressions- og klassifikationstræer

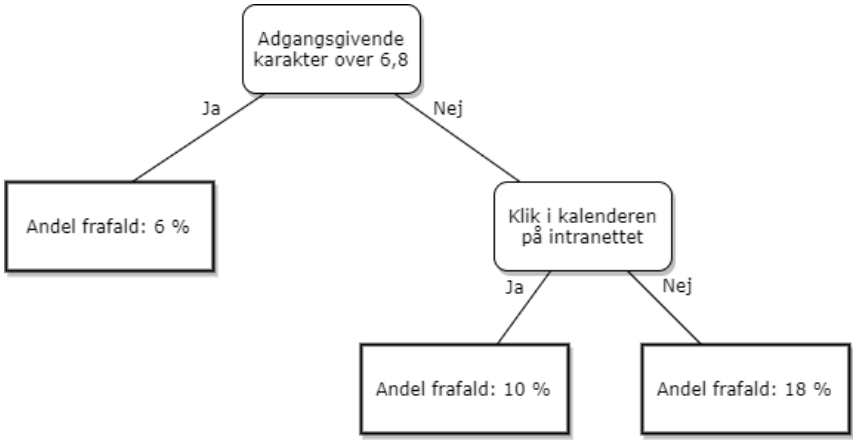
Regressions- og klassifikationstræer

Algoritmens logik

- Kan forstås som segmentering af data pba. ja/nej-spørgsmål
- Spørger: Hvilken variabel `j` skal jeg splitte ved hvilken værdi `s` for at mindske loss'et mest muligt?
- Eksempel: Er den studerende en kvinde? Er den over 18 år? 19 år? 20 år? …
- Loss'et er et mål for purity, dvs. et ens mønster i `Y`
- Binær, rekursiv splitting

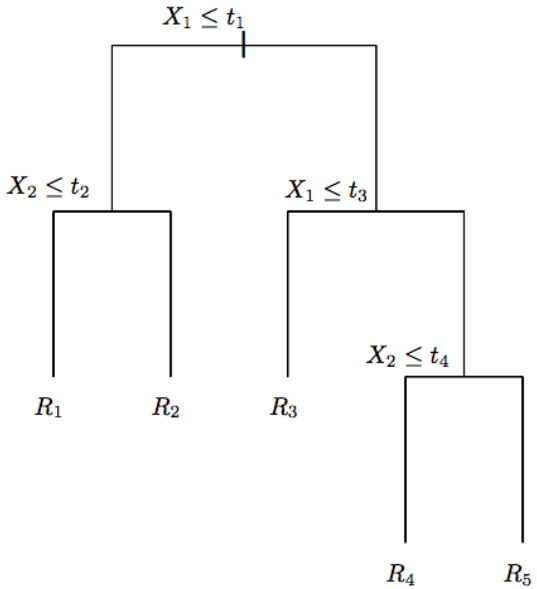
Regressions- og klassifikationstræer

Eksempel



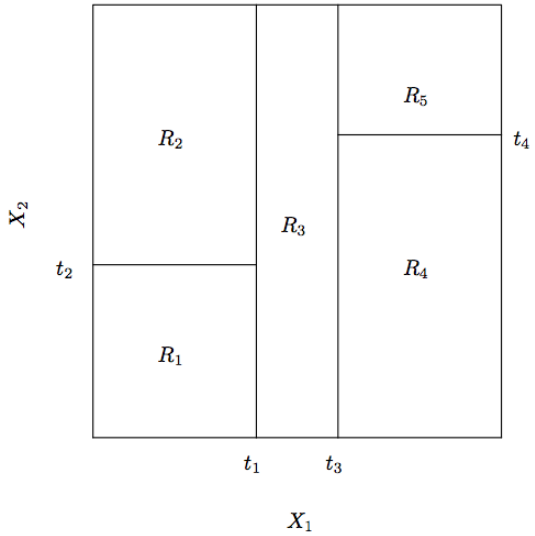
Regressions- og klassifikationstræer

Illustration I



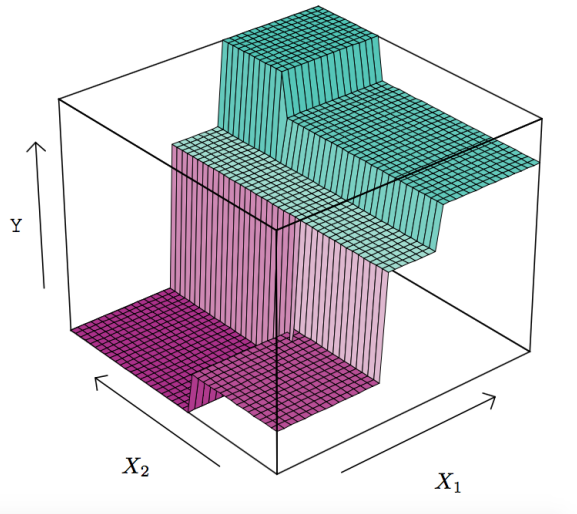
Regressions- og klassifikationstræer

Illustration II



Regressions- og klassifikationstræer

Illustration III



Regressions- og klassifikationstræer

Egenskaber og karakteristika

- Er *topdown* og *greedy*
- Fortsætter indtil et stopkriterium er nået, fx min. antal obs. i et blad
- Tidligere segmenteringer betinger senere segmenteringer
- Tillader komplekse interaktioner og non-lineariteter
- Er nonparametrisk og forudsætter ikke forhåndskendskab til DGP

Regressions- og klassifikationstræer

Loss-funktioner

$$\ell_{tree} = \arg\min_{\ell} \left[\sum_{x_i \in R_1(j,s)} \ell(y_i, \hat{y}_{R_1}) + \sum_{x_i \in R_2(j,s)} \ell(y_i, \hat{y}_{R_2}) \right]$$

Hvor $R_1(j,s) = \{X | X_J > s\}$ og $R_2(j,s) = \{X | X_J \leq s\}$ $

Loss'et:

- Ifm. regression et mål for error, fx MSE
- Ifm. klassifikation et mål for purity, fx 1 - error rate eller gini index

Regressions- og klassifikationstræer

Regularisering

Træer kan fitte data meget præcist => stor risiko for **overfitting.**

Det håndterer vi med regularisering, også kaldet **pruning**:

$$\ell_{tree} = \arg\min_{\ell} \sum_{m=1}^{M} \sum_{x_i \in R_m} \ell(y_i, \hat{y}_{R_m}) + \alpha M$$

Hvor M angiver antallet af blade i træet.

Minder om ridge- og LASSO-regression, som vi stødte på i sidste uge.

Først groes træet dybt, derpå beskæres det.

Regressions- og klassifikationstræer

Tuning

Tunings-parametre:

- `maxdepth`: træets maksimale dybde
- `minbucket`: min. antal obs. i et blad
- `cp`: complexity pruning, prisen for kompleksitet (antal blade)

Regressions- og klassifikationstræer

Fordele og ulemper

Fordele:

- **Intuitive**: Nemme at forstå (nemmere end lineær regression!)
- **Fleksible**: Kan nemt modellere komplekse interaktioner og non-lineariteter
- **Præ-processering**: Kræver mindre præproc., fx mht. skalering og outliers
- **Fitting**: Kan fitte data meget præcist

Ulemper:

- **Fitting**: Lav robusthed og tilbøjelighed til overfitting
- **Performance**: Typisk relativt lav prædiktiv performance

Random Forest

Random Forest

Karakteristika

- Ensemble-metode: Træner ikke træner ét, men en hel skov af træer
- Afholder majoritets-afstemning
- Bygger på *bagging* (bootstrap aggregation)

Random Forest

Bagging

Bagging:

- Hvis n uafhængige observationer $Z_1, \ldots, Z_n$ har variansen σ^2, har deres gennemsnit $\bar{Z}$ variansen σ^2/n. Det udnyttes i bagging.
- Træk samples fra træningssættet => fit flere modeller => beregn gns. af $\hat{y}$
- Træerne groes dybt (reducerer bias) og gns. findes (reducerer variansen)
- Udfordring: Træerne vil være korrelerede, især hvis der er stærke prædiktorer

Random forest

- Løsning: Med RF samples yderligere prædiktorerne til rådighed i et split
- Typisk værdi: $\sqrt{p}$

Random Forest

Tuning

- Kan i princippet tunes som alm. træer, fx `maxdepth`
- Kan også tunes på ensemble-niveau i form af `ntrees`
- I praksis kun nødvendigt at tune `mtry`: antal prædiktorer til rådighed ved splits

Random Forest

Fordele og ulemper

Fordele:

- Ofte rigtig god performance
- Forståelig algoritme
- Har ikke tendens til overfitting (fordi træerne er uafhængige)

Ulempe:

- Sværere at tolke variables betydning ift. ét træ (dog: importance-mål)
- Sværere at visualiserede end ét træ

Gradient Boosted Trees

Gradient Boosted Trees

Karakteristika

- Ensemble-metode ligesom RF
- Baseret på *boosting*: sekventiel fitting af træer
- Er kendt for fremragende performance

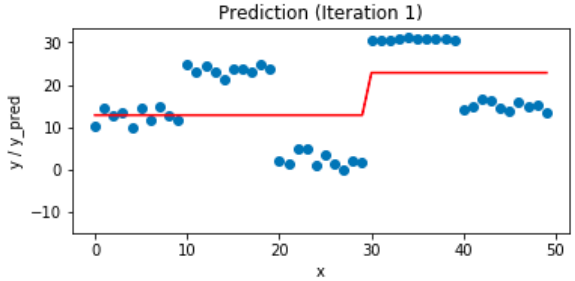
Gradient Boosted Trees

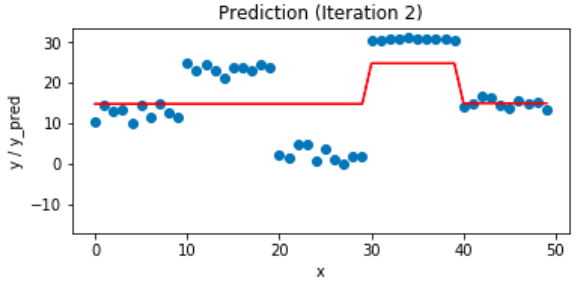
Boosting

- **Logik som i bagging**: Vi kombinerer et ensemble af træer
- **Forskel**: Træerne fittes sekventielt til tidligere træers residulaler
- **Loss-funktion**: Vi fitter altså til residualerne snarere end til selve Y
- **Idé**: Vi forbedrer langsomt modellen der, hvor den er dårligtst
- **Intuition**: Vi modellerer residualerne indtil de varierer tilfældigt
- **Udfordring**: Booster vi for meget fitter vi til støjen (overfitting)

Gradient Boosted Trees

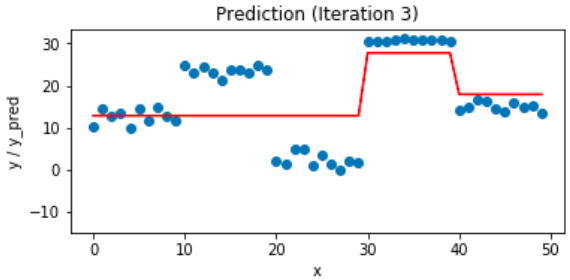
Illustration af boosting I

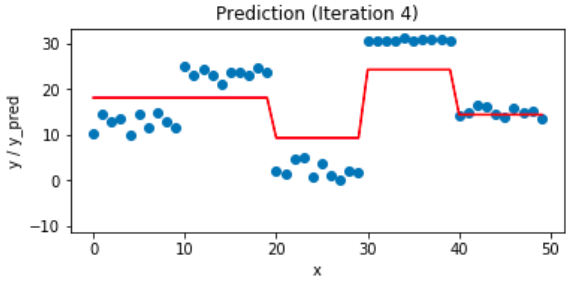




Gradient Boosted Trees

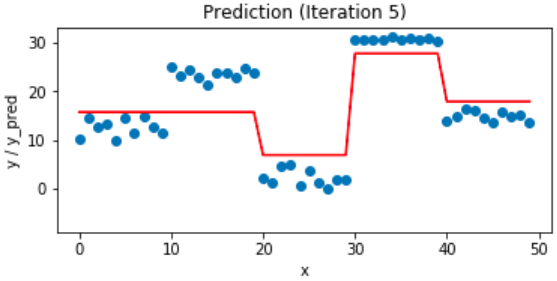
Illustration af boosting II

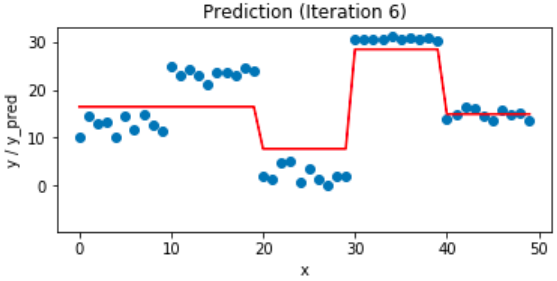




Gradient Boosted Trees

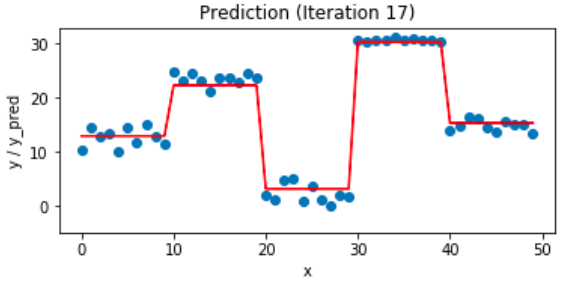
Illustration af boosting III

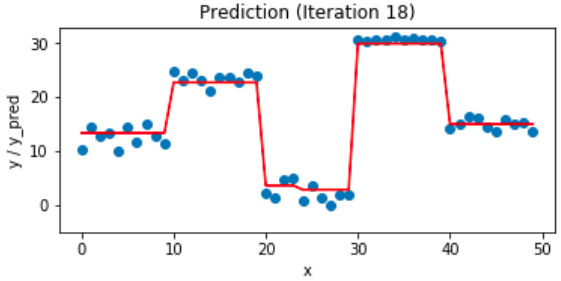




Gradient Boosted Trees

Illustration af boosting IV





Gradient Boosted Trees

Tuning

Parametre vedr. træet:

- `max_depth`: dybde (ofte fitter vi bare stubbe, dvs. `max_depth` = 1)
- `min_child_weight`: min. antal obs. i hvert blad
- `gamma`: parameter som straffer kompleksitet

Parametre vedr. sampling (tilføjer variation som i RF):

- `colsample_bytree`: antal prædiktorer tilgængelige ved split
- `subsample`: andelen af observationer hvert træ fittes til

Parametre vedr. boostingen:

- `eta`: læringsrate (typisk god performance v. langsom læring)
- `nrounds`: antal boosting-iterationer (invers ift. `eta`)

Gradient Boosted Trees

Fordele og ulemper

Fordele:

- Ofte særdeles god performance

Ulempe:

- Langsommere og mere omstændig at tune end fx RF
- Sværere at forstå end fx RF

caret

`caret`

Introduktion

Hvad er `caret`?

- En imponerende pakke i R til at træne maskinlæringsmodeller end-to-end

Typisk tilgang:

1. Præ-processering
2. Definere controls, såsom resampling-metode
3. Træne og tune model
4. Evaluere model

caret

Eksempel i R

Workshop

Workshop

Challenge:

- Byg en bedre prædiktionsmodels end din underviser!
- Hvem får den højeste AUC?

Find opgaverne på Github under PDS/opgaver/:

- `11_opgaver.R`

Afrunding

Challenge

Hvor høj en AUC opnåede I?

Ønsker til sidste lektion

Inputs

- Recap af emner fra pensum?
- Kun workshop og øvelser?
- Feedback på 1 pager eksamensoplæg?
- Opponering på seminaropgaver i klynger?
- Introduktion til R Markdown og Shiny?
- Introduktion til LaTeX?
- Se film?
- …

Næste gang

- Indhold:
  - Usuperviseret læring
- Pensum:
  - ISL: kap 10 afs 10 - 10.3
- DataCamp:
  - Unsupervised Learning in R

Boblejagten

- **KPI**: Mest XP
- **Periode**: 6. feb. - 6. maj
- **Who's awesome?**

Tak for i dag!