Political Data Science

Lektion 12: Usuperviseret læring

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
   - Reduktion af dimensionalitet med PCA
   - Clustering med K-means
   - Clustering med hierchical clustering
3. Workshop
4. Afrunding og næste gang (og sidste 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

Opsamling fra sidst

Opsamling fra sidst

Øvelser

Opgave-script indeholder "løsning" på challengen.

Opsamling fra sidst

Træ-modeller

Hvad er logikken i regressions- og klassifikationstræer?

Hvad er de centrale forskelle på singulære træer og de to ensemble-metoder Random Forest og Gradient Boosted trees?

Snak med dine sidemakkere.

Usuperviseret læring

Usuperviseret læring

Introduktion

Hvad er usuperviseret læring?

- Et sæt af statistiske redskaber når vi ikke har et outcome Y, men kun uafhængige varible $X_1, X_2, \ldots X_p$ målt på n observationer

Formålet med usuperviseret læring

- Eksplorativ analyse af $X_1, X_2, \ldots X_p$
- Redskab til at finde og visualisere mønstre i data
- Redskab til at idenficicere sub-grupper i data

Usuperviseret læring

Eksempler

- Sub-grupper i gen-data, kundedata, vælgerdata
- Detektion af anomaliteter, fx hos NETS eller SKAT
- Topic models til at gruppere tekster efter emner
- Alliancemønstre i FT-afstemninger (supplerende pensum)
- Ansigtsgenkendelse på facebook-billeder

Usuperviseret læring

Udfordringer

- Ingen prædeficerede mål => subjektiv analyse
- Ingen universel validerings-metode => svært at bedømme resultater
- Ingen "korrekte" resultater

Principal Components Analysis (PCA)

Principal Components Analysis

Introduktion

Hvad er det?

- En usuperviseret tilgang til at finde et fåtal af dimensioner i et datasæt, der indeholder det meste af datasættets information

Principal components-analyse:

- Er usuperviseret fordi vi kun har $X_1, \ldots, X_p$ og ingen Y på forhånd
- Refererer til processen 1) at finde PCs, 2) bruge PCs til at forstå data

Principal Components Analysis

Brug af PCA

Hvornår bruger vi PCA?

- Antag vi har et datasæt med 10 variable, fx om politiske holdninger, som i forskellig grad korrelerer med hinanden. Antag vi bliver bedt om at finde frem til og kommunikere de mest interessante mønstre i datasættet. Hvordan griber vi opgaven an?
- Vi kan visualisere korrelationerne med p * (p - 1) / 2 scatterplots. Når p = 10 som ovenfor giver det 45 scatterplots. Er p = 20 er der 190 scatterplots.
- Det er uoverskueligt og de fleste plots vil være uinformative.
- Med PCA kan vi simplificere og visualisere mønstrene i data.

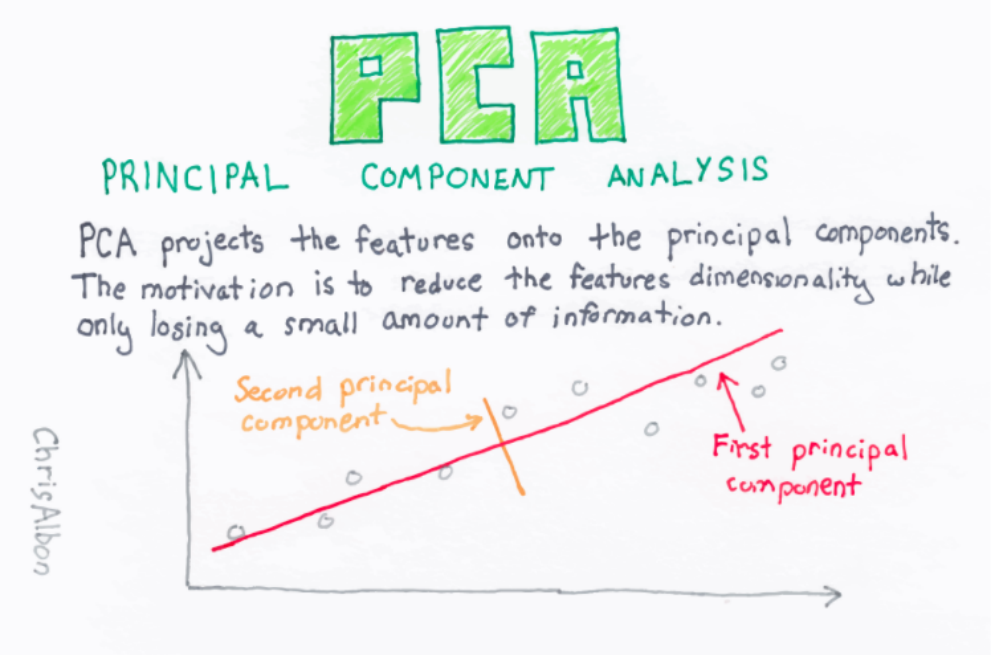
Principal Components Analysis

Koncept

- Idéen er, at hver observation n lever i et p-dimensionalt rum.
- Ikke alle dimensioner er lige interessante
- Med PCA søger vi et lille antal dimensioner, der er så interessante som muligt.
- "Interessant" = variation.

Principal Components Analysis

Konceptet illustreret



Principal Components Analysis

Algoritmen

Hver PC er en lineær kombination af p variable:

$$PC_1 = \phi_{11}X_1 + \phi_{21}X_2 + \cdots + \phi_{p1}X_p$$

Algoritmen maksimerer variansen i PC_1 på tværs af forskellige værdier for ϕ.

PC_1 er den **normaliserede**, lineære kombination af variable med **størst varians**.

PC_2 har næststørst varians, og er desuden othogonal på den første - osv.

Udtrykket er normaliseret så $\sum_{j=1}^{p} \phi_{j1}^2 = 1$, fordi en arbitrært stor loading ville give arbitrær stor varians, ligesom de enkelte variable er normaliserede.

Elementerne $\phi_{11}, \ldots, \phi_{p1}$ kaldes for PC'ens **loadings**.

Principal Components Analysis

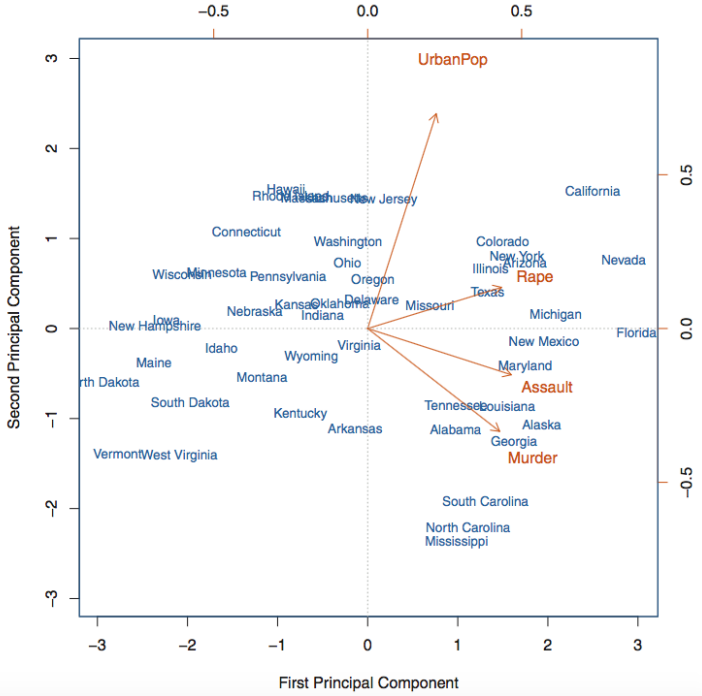
Eksempel: PCs og loadings

PCA er udført på datasættet `USArrest`, der indeholde 4 uafhængige variable.

	PC1	PC2
Murder	0.5358995	−0.4181809
Assault	0.5831836	−0.1879856
UrbanPop	0.2781909	0.8728062
Rape	0.5434321	0.1673186

Principal Components Analysis

Eksempel: PCs og loadings



Principal Components Analysis

Normalisering

Variablenes gennemsnit og enhed har betydning for deres varians. Det er fx ikke ligegyldigt, om vi måler tyverier pr. 1000 eller 100.000 indbyggere.

Vi vil derfor gerne normalisere data:

- **centrering**: variablene skal centreres så gns. = 0
- **skalering**: ofte skal variablene skaleres så sd = 1

Principal Components Analysis

Evaluering af PCs: Proportion of variance explained

Hvor meget forklaringskraft har vores principal components?

Det kan vi udrykke ved PVE: proportion of variance explained.

Det er én PC's varians dels med alle PC'ernes samlede varians.

Det kan vi beregne for hver PC ved at kvadrere PC'ens standardafvigelse og dele det med summen af den kvadrerede standardafvigelse for alle PCs.

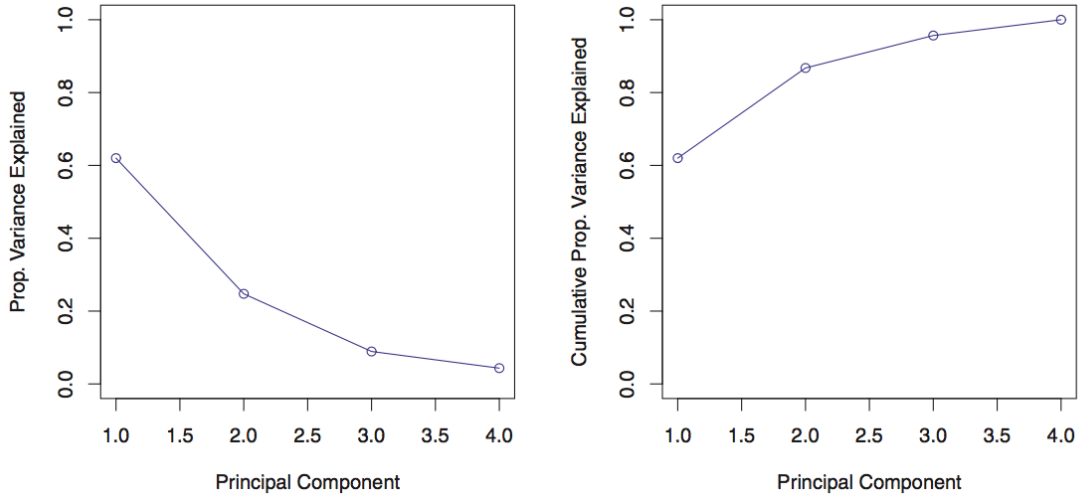
R kan også spytte VE, PVE og kumulativ PVE for hver PC ud for os.

Principal Components Analysis

Hvor mange PCs har vi brug for?

Svaret findes ikke a priori. Mål: maksimere information og minimere antal PCs.

Vi kan informere beslutningen ved at kigge efter en **elbow** i et **scree plot**:



Clustering

Clustering

Introduktion

Hvad er det?

- Et bredt sæt af teknikker til at finde subgrupper i data.
- Hver gruppe er så internt ens og eksternt forskellig som muligt.
- Grupperne er udtømmende og ikke-overlappende.

Formål

- Simplificere data ved færre, sigende informationer i form af subgrupper.

Metoder vi vil fokusere på:

- K-Means clustering
- Hierarchical clustering

K-Means clustering

K-Means clustering

Karakteristika

- **Ide**: en god cluster har så lille within-cluster variation som muligt.
- **Algoritme**: minimere variationen within-clusters på tværs af clusters.
- **Definition af variation**: Den mest udbredte er Euklidisk afstand.
- **Euklidisk afstand**: måler afstand mellem to punkter i et p-dimensionalt rum.
- **K-Means**: Minimerer parvis Euklidisk afstand ml. alle punkter i hvert cluster.
- **Optimering**: Enormt komplekst optimeringsproblem!

K-Means clustering

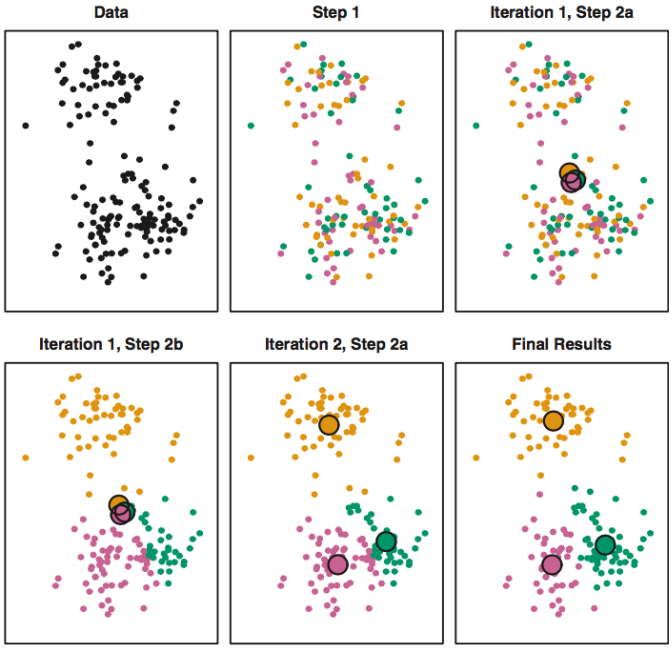
Algoritmen

Vi løser i stedet optimeringsproblemet med tilnærmning.

1. Bestem et antal clusters
2. Tilskriv hver observation et tal fra 1, …, K som første cluster
3. Fortsæt indtil cluster-fordelingen ikke ændrer sig med at:
   - Beregn hver clusters centroid (en vektor med varibel-gns.)
   - Tilskriv hver observation til den nu nærmeste centroid.

K-Means clustering

Illustration



K-Means clustering

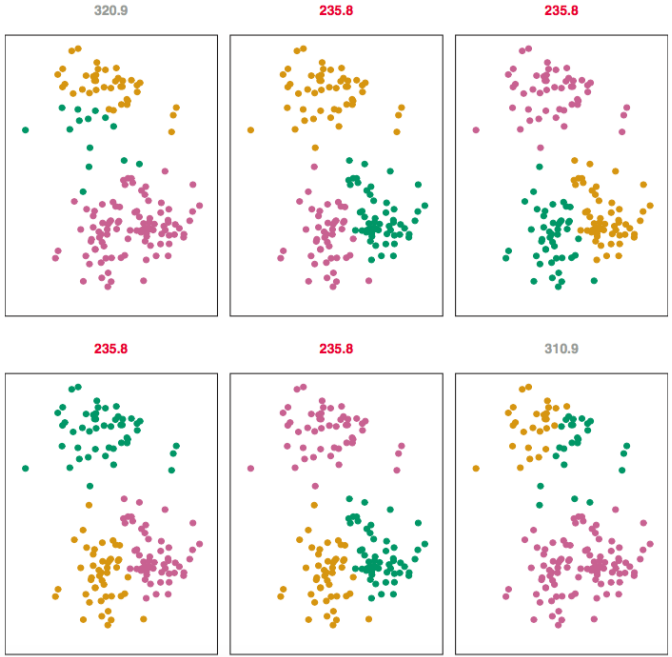
Udfordringer

Især to udfordringer står tilbage:

1. Algoritmen K-Means er kun garanteret at finde et lokalt optimum.
2. Hvordan præspecificerer vi antallet af clusters?

K-Means clustering

Optima



Hierarkisk clustering

Hierarkisk clustering

Introduktion

Sammenlignet med K-Means gør især to ting hierarkiske clustering attraktiv:

1. Med HC behøver vi ikke specificere antal clusters på forhånd.
2. Med HC producerer vi et træ-lignende **dendrogram**, der visualiserer clusters for k = 1, … n samtidig.

Hierarkisk clustering

Algoritme I

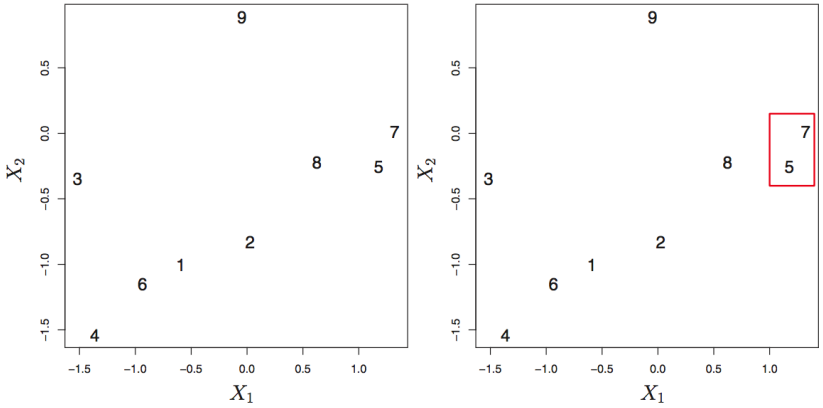
Konceptet hierarkisk clustering består i at bygge et træ, oftest bottom up.

Algoritme:

1. Start med at lade hver observation n være sin egen cluster.
2. Beregn de parvise forskelle mellem alle clusters (fx Euklidisk afstand)
3. Så længe antal clusters > 1:
   - Slå de to clusters sammen, der er mindst forskellige.
   - Beregn de nye parvise forskelle.

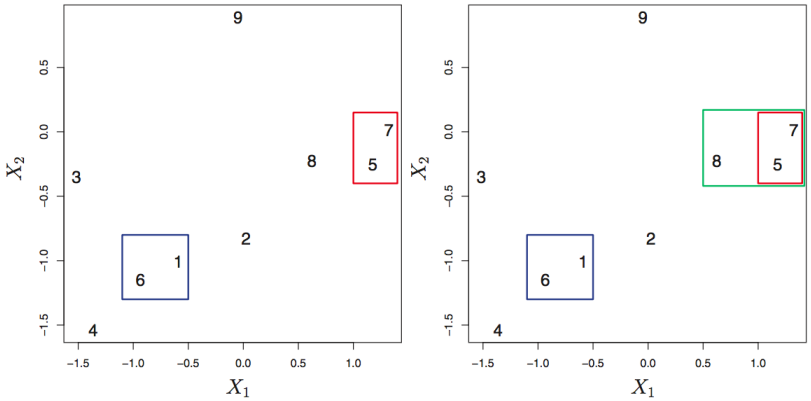
Hierarkisk clustering

Algoritme II



Hierarkisk clustering

Algoritme III



Hierarkisk clustering

Linkage

Hvordan måler vi forskellighed, når clusters indeholder mere end 1 observation?

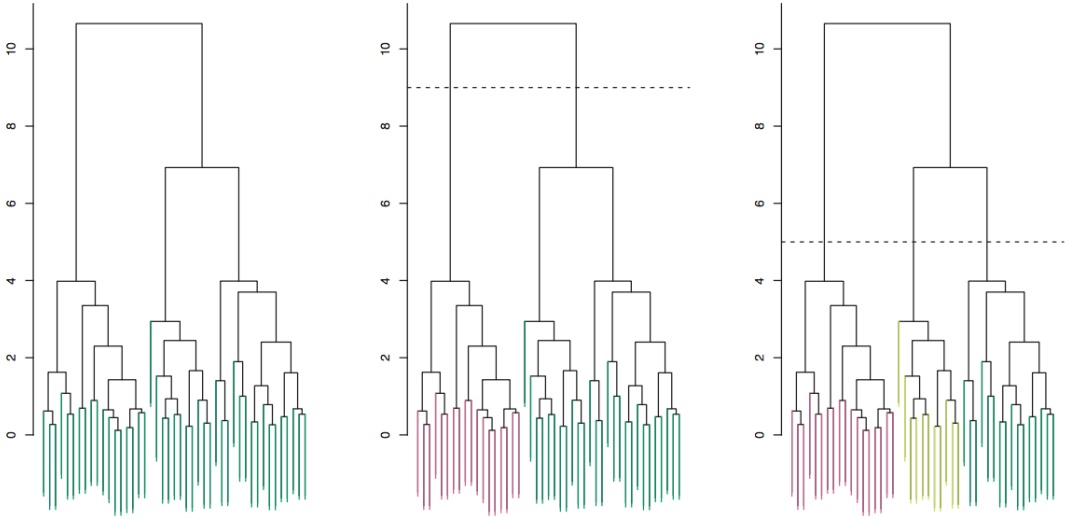
Det gør vi med **linkage**:

- **Complete**: Maksimal parvis afstand ml. observationerne i cluster A og B
- **Single**: Minimal parvis afstand ml. observationerne i cluster A og B
- **Average**: Gns. parvis afstand ml. observationerne i cluster A og B
- **Centroid**: Afstanden ml. centroids i cluster A og B

Typisk foretrækker vi **complete** og **average**, fordi de oftest giver de mest balancerede clusters.

Hierarkisk clustering

Dendrogram



Hierarkisk clustering

Fortolkning af dendrogrammer I

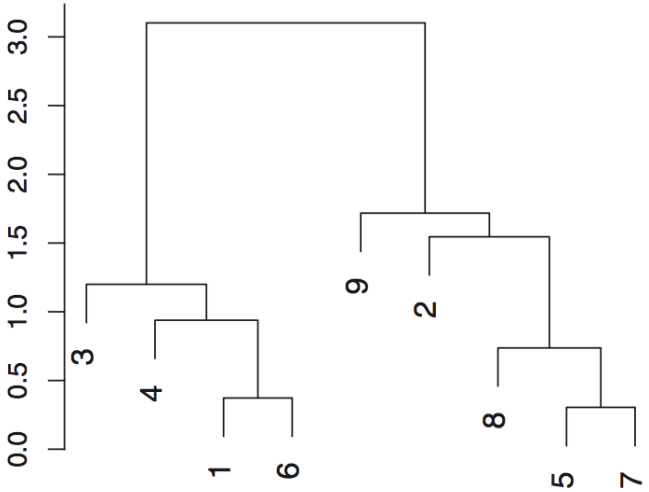
Højden i dendrogrammer:

- Højden på en fusion ml. clusters indikerer deres forskellighed (vertikal akse)
- Bredden fortolker vi ikke på (horisontal akse)

Hierarkisk clustering

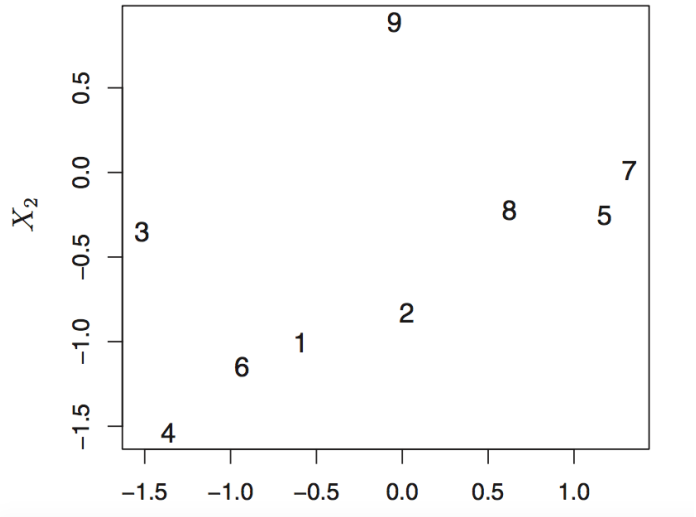
Fortolkning af dendrogrammer II

Hvad kan vi fortolke om observationernes forskellighed her?



Hierarkisk clustering

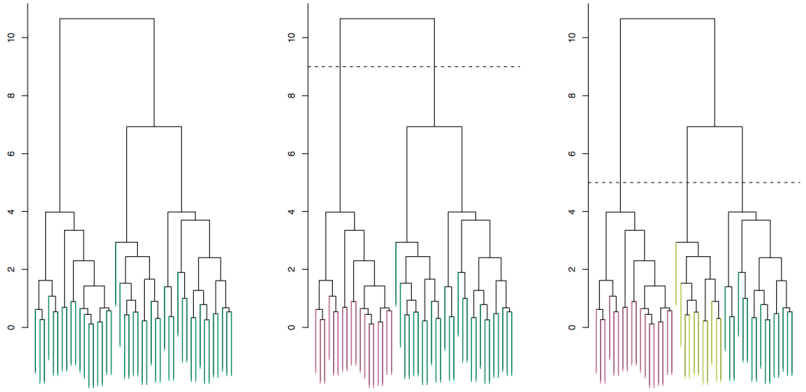
Fortolkning af dendrogrammer III



Hierarkisk clustering

Antal clusters

Vi sætter antal clusters ved at lave et cut i dendrogrammet, typisk baseret på højden af fusionerne og eventuelle ønsker til antal clusters.



Hierarkisk clustering

Ulemper

Hierarkisk clustering bygger på en antagelse om en nested struktur i clusters:

- Antag vi har et datasæt om respondenter fra Danmark, Tyskland og Frankrig, og lige mange fra hvert land er hhv. mænd og kvinder. Her kan det bedste split i to grupper være i køn, mens det bedste split i tre grupper kan være i nationaliteter. Her er grupperingerne ikke nestede - den bedste opdeling i tre grupper fås ikke ved at opdele den én af to grupper i to.

I en situation som den vil K-Means typisk gruppere data mere præcist end HC.

Clustering i praksis

Processen

1. Præprocessering:
   - Hvis variablene er forskellige enheder er det som regel en god idé at normalisere data så gns. = 0 og SD = 1. Normalisering skader sjældent.
2. Fastsættelse og afprøvelse af parametre:
   - **K-Means**: Antal clusters
   - **Hierarkisk clustering**: Type forskellighed + linkage + cutoff
3. "Validering":
   - Hvor robuste er de fundne clusters mod ændringer i parametre?
   - Hvor robuste er de fundne clusters mod ændringer i data, fx subsetting?
   - Forekommer de fundne clusters meningsfulde og informative?

Husk på: Der findes hverken en naturgiven tilgang eller absolut sande clusters - vi leder blot efter subgrupper, som kan gøre os klogere!

Workshop

Workshop

Find opgaverne på Github under PDS/opgaver/:

- `12_opgaver.R`

Afrunding

Sidste lektion

Inputs

Lektion 1:

- Optegning af overordnede linjer i kurset
- Further reading
- Boblejagt
- Evaluering

Lektion 2:

- Mini-oplæg og opponering på 1-pagers om seminaropgaver i klynger
- Kort feedback på 1-pagers

Næste gang

- Indhold:
  - Refleksioner om data science
- Pensum:
  - Anderson (2008),
  - Lazer et al (2014),
  - Samii (2016),
  - Athey (2017),
  - Mittelstadt et al (2016),
  - Hofman et al. (2017)
- DataCamp:
  - [Valgfrit kursus]

Tak for i dag!