Data Science for Bibliotekarer

December 29, 2017

Indhold

1 Indledning

TODO

- expand

Formålet med denne bog er, at give dig en smagsprøve på hvad data science er, og en fornemmelse af hvordan man leger med data. Dette gør vi, ved at starte med en introduktion til værktøj, og hvad data science er, - og derefter dykker vi ned i nogle real-world case studies.

Første trin er at komme *i gang med Jupyter*, som er er et værktøj til at lave data science i Python. Denne bog er skrevet som en række af Jupyter notesbøger, og dette kapitel sætter dig i stand til at køre, og eksperimentere med eksemplerne.

Derefter undersøger vi: *Hvad er data science*, hvilket giver det teoretiske fundament for at forstå tilgangen i eksemplerne.

En introduktion til *Python programmering*, giver det praktiske fundament for at kunne læse koden i eksemplerne.

Herefter kommer de egentlige case studies.

Og til slut afrunder vi med konklusion, og foreslag til videre studier.

1.1 Målgruppe

Denne bog er skrevet til en workshop om data science for bibliotekarer, så den forudsætter ingen forkendskab til programmering. Ligeledes er eksemplerne taget fra biblioteksverdenen.

1.2 Læringsmål

Du bliver i stand til at

- køre python *notebooks*, og lave små ændringer / eksperimenter i disse
- ...

2 I gang med Jupyter

TODO
Guide to jupyter after install

2.1 Opsætning

Jupyter Notebook er det værktøj, som vi vil bruge til at eksperimentere me data science. Dette kan enten installeres lokalt på din egen computer, eller du kan køre det i skyen. Begge dele beskrives herunder.

2.1.1 Lokal installation

Den letteste måde at installere Jupyter Notebook lokalt på din maskine er via af værktøjet Anaconda:

- Åbn `https://www.anaconda.com/download` i din webbrowser.
- Download og kør installationsprogrammet for `Python 3.?` versionen af Anaconda, som passer til din computer (Linux/Windows/Mac, 32bit/64bit). Følg eventuelt vejledningen på websiden.
- Start *Anaconda Navigator*, som nu er installeret, og vælg derefter at starte / "Launch" Jupyter Notebook.

Dette starter Jupyter, og åbner en webbrowser, hvor du kan se filerne på din maskine, og redigere/køre Notebooks. Næste trin er at hente eksemplerne, så du selv kan lege med dem:

- Åbn `https://github.com/solsort/data-science-for-bibliotekarer/releases` i din webbrowser.
- Klik på / download *Source code (zip)* for den nyeste release, og udpak denne på din computer.

Når filerne er pakket ud, kan du igen vende tilbage til Jupyter Notebook i din webbrowser. Her kan du vælge den mappe hvor du har pakket `data-science-for-bibliotekarer` ud, og begynde at køre Notebooks.

2.1.2 Kørsel i skyen

Azure giver mulighed for at køre Jupyter Notebooks gratis i skyen. For at kunne køre eksemplerne, skal du gøre følgende:

- Åbn https://notebooks.azure.com i din webbrowser.
- Klik på `Sign In`, opret bruger og log ind.
- Klik på `Libraries`, og derefter `New Library`
  - Vælg `From GitHub`
  - Skriv `solsort/data-science-for-bibliotekarer` der hvor der står *Git repo (required)*.
  - Find på et navn/id til projektet (*Friendly Name* / *Unique id* ...), - dette kan være hvad som helst
  - Klik på import (først muligt når repository, name og id er udfyldt)

Herefter vil eksemplerne blive loadet. Du har nu et kørende Jypyter Notebook environment, og kan åbne de enkelte notebooks ved at klikke på dem.

Næste gang du logger ind, kan du bare klikke på dit igangværende projekt, og behøver ikke at vælge `New Library` etc.

3 Hvad er data science

TODO

- restructure/rewrite

Data science handler om, at finde indsigt og viden ud fra data, gennem en videnskabelig tilgang. Det er en blanding af forskellige faglige felter, her iblandt:

- Programmering og datalogi: Hvordan vi instruerer computeren i at finde viden fra data.
- Informationsvidenskab: Hvordan vi strukturerer og formidler den viden vi får ud af data.
- Statistik og matematik: Hvordan vi konceptuelt finder viden i data.

Der er også en del overlap med de dele af datalogien, som hedder maskinlæring og data mining, som handler om hvordan man kan få computeren til (selv) at finde mønstre i data.

3.1 Strukturerede data

Data er alt hvad vi kan skrive ned. Eksempler på data er: 1) observationer, såsom en note hver gang en bog bliver udlånt, 2) tekst, såsom de fleste felter i metadata. Kvalitative data er data der kræver menneskelig fortolkning, - det kan være interviews, beskrivelse af bøger, etc. Kvantitaive data er data som en computer direkte kan arbejde med, - det kan være statistiske data, lister af hændelser med tidspunkt, etc.

Vi vil arbejde med forskellige *typer* af data:

- Tal, som både kan være heltal (... `-2`, `-1`, `0`, `1`, `2`, ...), eller kommatal (i.e. `-17.42`, `0.12345`, `1001.37`).
- Tekststrenge eller strenge, som er kortere eller længere stykke tekst. Ordet streng kommer af at tekst er en række/sekvens/tråd/streng af enkelte bogstaver. Vi vil ofte skrive eksempler på tekststreng i anførselstegne, i.e. `"dette er en streng"`.
- Sandhedsværdier eller logiske værdier, som har blot to mulige værdier: *sandt* (`true`) eller *falskt* (`false`).
- Lister af data, som kan indeholder strenge, tal, sandhedsværdier, og også selv indeholde lister og ordbøger. Vi vil ofte skrive lister med firkantede paranteser, eksempel: `["Høeg", "Murakami", "Blixen"]`.
- Opslagsværker eller ordbøger/dictionaries, som giver mulighed for at slå en tekststreng op og få data tilbage. Et eksempel på en ordbog kan være metadata for en bøger, hvor man eksempelvis kan slå "forfatter" og "titel" op, og måske får "A. A. Milne" og "Peter Plys" tilbage. Ordet/strengen, som man slår op i ordbogen kaldes ofte nøglen, og de data man får tilbage kaldes værdien. Ordbøger skrives ofte med krøllede paranteser, så

ordbogen med metadata kunne skrives som: `{"titel": "Peter Plys", "forfatter": "A. A. Milne"}`.

Ud fra disse datatyper, kan vi nu strukturere data, eller lave *datastrukturer*. Her er et eksempel på metadata for nogle bøger:

```
[{"titel": "Peter Plys",
  "forfatter": "A. A. Milne",
  "nøgleord": ["børnebog"],
  "tilgængelige": true,
  "udlån": 4312},
 {"titel": "Turen går til Berlin",
  "forfatter": "Kirstine Therkelsen",
  "nøgleord": ["Berlin", "rejsefører"],
  "tilgængelige": false,
  "udlån": 329}]
```

Denne måde at skrive data kaldes *JSON*, og er en af de mest almindelige måde at skrive strukturerede data. Eksemplet indeholder en liste af ordbøger, som igen kan indeholde andre datatyper. Bemærk også data, skrevet herover som JSON, faktisk er tekst, hvilket igen kunne repræsenteres som en tekststreng.

3.2 Regnemaskiner og tal

Computere er kun regnemaskiner. Data er kun tal. Dette lyder måske svært at tro, men det er sandt.

Bogstaver i computeren er tal: `A` er 65, `B` er 66, `C` er 67, `D` er 68, `E` er 69, `F` er 70, `G` er 71, `H` er 72, `I` er 73, `J` er 74, `K` er 75, `L` er 76, `M` er 77, og så videre, og mellemrum, `" "` er 32. Hvad tror du, at disse tal betyder: `72 69 74 32 77 69 68 32 68 73 71`.

Husk fra foregående afsnit, at data og datastrukturer kan skrive som en tekststreng. Vi har nu set at bogstaver, og derved tekst, kan skrives som tal. Alle data kan derfor skrives som tal.

4 Python Programmering

TODO

What is programming

- Recipe, instructions for computer
  - computer = forvokset regnemaskine, kan kun gøre hvad den bliver bedt om
  - step by step instruction, computer is stupid, only does excatly what told
- Example of programming languages easily available
  - JavaScript
  - Python
  - Shell
- Getting started with Python/jupyter-notebooks
  - running within notebooks.azure.com
  - running locally with https://www.anaconda.com/download

What is data

- Concrete/paper examples: numbers, text, checkmark/questionmark, lists, indexes
- Structured data: JSON
  - example/exercise bibliographical data
- Numbers, Matrices/vectors
- Data types
  - Numbers
  - Strings
  - Lists
  - Dictionaries
  - Matrices/vectors
  - ... and many morre
- What is data science
  - "datalogi" - computer science vs. data science
  - statistics and machine learning
    - unsupervised
      - example: k-means

        * supervised
            - example: statistical

Smagsprøver på python

- forklaring af konstruktioner brugt i eksemplerne

5

Case: Genre-rum

TODO

- øvelser

Vi har en idé om, hvad *afstanden* er mellem to punkter. Hvis vi har to prikker på et stykke papir, så véd vi, hvordan vi måler afstanden mellem dem. Hvis vi har to steder i vores dagligstue, så har vi også en idé om deres afstand.

Hvordan kan vi tale om afstand mellem to bøger? Vi vil nok forvente, at bøger indenfor samme genre ligger tæt på hinanden, eksempler:

- Afstanden mellem "Peter Plys" og "Frøken Smillas fornemmelse for sne" er nok større end afstanden mellem "Cirkeline" og "Cykelmyggen Egon".
- En rejsebog om Berlin er nok tættere på en rejsebog om Paris, end på en håndarbejdsbog.
- Bøger af samme forfatter befinder sig nok i nærheden af hinanden.

Forestil dig, at vi har et genre-rum, hvor der er et punkt for hver eneste bog, og vi kan måle afstanden mellem dem. Dette vil give nye muligheder for at gå på opdagelse i litteraturen. Anbefalinger skabes ved at finde de nærmeste nabopunkter. En genre består af punkter i nærheden af hinanden. Dette gør, at vi kan bruge computeren til at udforske litteraturen.

På papir, i vores dagligstue såvel som i genrerummet, kan afstand defineres matematisk således: Hvis vi har to punkter, a og b, så er afstanden mellem dem $\sqrt{(a-b)^2}$. I én dimension er det afstanden mellem to tal. Eksempel: afstanden mellem 1 og 3 er 2, hvilket kan udregnes som $\sqrt{(1-3)^2} = \sqrt{(-2)^2} = \sqrt{4} = 2$. I to dimensioner er det afstanden mellem to prikker på et stykke papir, også kaldet den "Euklidiske afstand", $\sqrt{(x_a - x_b)^2 + (y_a - y_b)^2}$. Eksempel: afstanden mellem koordinaterne $(1, 2)$ og $(4, 6)$ er 5 hvilket vi kan udregne som $\sqrt{(1-4)^2 + (2-6)^2} = \sqrt{3^2 + 4^2} = \sqrt{9 + 16} = \sqrt{25} = 5$. I tre dimensioner er det afstanden mellem to punkter i rummet, i.e. $\sqrt{(x_a - x_b)^2 + (y_a - y_b)^2 + (z_a - z_b)^2}$, - og det fortsætter på samme måde i fire, fem, seks, ... dimensioner.

I matematik kalder vi ofte koordinaterne for *vektorer*. Eksempelvis er en 5-dimensionel vektor, blot en liste af fem tal, og en 100-dimensionel vektor er en liste af hundrede tal.

Hvordan skaber vi et sådan genrerum? Vi har en masse statistik om lån på bibliotekerne. Hvis vi antager, at man ofte låner indenfor samme genre, så kan computeren ud fra disse data udregne et genrerum.

Jeg har udregnet et sådan genrerum for 10.000 biblioteksmaterialer, og i det følgende vil vi undersøge, om afstanden mellem bøger i genrerummet giver mening.

5.1 Eksperimenter med afstand mellem bøger

Når vi laver et program, må vi først fortælle computeren, hvilken funktionalitet vi har brug for: `bibdata` indeholder bibliografiske data, og genrerummet, som jeg har beregnet. `numpy` indeholder matematikfunktionalitet:

```
In [1]: import bibdata
        import numpy
```

I `bibdata` er der en funktion, som returnerer titel/forfatter, hvis vi kommer med nummeret på et biblioteksmateriale:

```
In [2]: bibdata.title_creator(8955)
```

```
Out[2]: 'Peter Plys : komplet samling fortællinger og digte - A. A. Milne (book)'
```

Herover ser vi at bog nummer 8955 er "Peter Plys". Ligeledes kan vi se de øvrige bøger, som vi vil eksperimentere med herunder:

```
In [3]: [(book_number, bibdata.title_creator(book_number))
           for book_number in [8955, 8214, 616, 580, 149, 278, 126, 29, 688]]
```

```
Out[3]: [(8955,
          'Peter Plys : komplet samling fortællinger og digte - A. A. Milne (book)'),
         (8214, 'Frøken Smillas fornemmelse for sne : roman - Peter Høeg (book)'),
         (616, 'Cirkeline bliver til - Hanne Hastrup (book)'),
         (580, 'Cykelmyggen Egon - Flemming Quist Møller (book)'),
         (149, 'Turen går til Berlin - Therkelsen Kirstine (book)'),
         (278, 'Turen går til Paris - Aske Munck (book)'),
         (126, 'Alt om håndarbejdes strikkemagasin -  (other)'),
         (29, '1Q84 - Haruki Murakami (audiobook)'),
         (688, 'Kafka på stranden - Haruki Murakami (book)')]
```

Disse kan derefter navngives, så de er lettere at arbejde med.

```
In [4]: peter_plys = 8944
        smilla = 8214
        cirkeline = 616
        cykelmyggen = 580
        berlin = 149
        paris = 278
        strikning = 126
        q84 = 29
        kafka = 688
```

Vi kan finde punktet i genrerummet for en bog via `bibdata.genres`. Selve genrerummet er 100-dimensionelt, så vektoren består af 100 tal. I programmering kaldes vektorer ofte for *arrays*.

```
In [5]: bibdata.genres[peter_plys]
```

```
Out[5]: array([ 0.00908454, -0.04033043, -0.01284507,  0.00788709,  0.01718174,
                0.00128158, -0.07500522, -0.01639324,  0.06427721, -0.0099812 ,
                0.00083797,  0.01363433, -0.00866813,  0.01482092,  0.10160233,
                0.08134543, -0.04255533, -0.00833438,  0.02536649,  0.02128514,
                0.00554724, -0.00080702,  0.01274936, -0.03601923,  0.01916139,
                0.02077907, -0.04609259, -0.08917641,  0.07050021,  0.01678492,
                0.01278447,  0.00108154,  0.01457724,  0.04591186, -0.01287235,
                0.0202236 ,  0.00102133,  0.14692573,  0.18963774,  0.08172592,
               -0.01773496,  0.11990478, -0.12269035, -0.33739865,  0.23169368,
                0.01569639, -0.06665396,  0.00139089,  0.07659317,  0.00650507,
               -0.01185874,  0.01587052,  0.07172235,  0.00525925,  0.11984863,
                0.10747883,  0.25288281,  0.07708268, -0.2325371 , -0.10132218,
                0.07604308,  0.05801751,  0.02945208, -0.03336021, -0.08060202,
                0.03639616,  0.02639744, -0.02000082, -0.02184357,  0.02158996,
                0.04069334,  0.00409769, -0.15620241, -0.14288074, -0.04555338,
               -0.0875617 , -0.50226035,  0.18508342,  0.19686877,  0.11195053,
               -0.18041552,  0.09638708,  0.02675614,  0.04779613,  0.06875322,
               -0.08385483, -0.03366541, -0.01312519,  0.14200541, -0.06419092,
               -0.03430816,  0.07344945, -0.01200862,  0.01030531,  0.01404084,
                0.01365915,  0.04134672,  0.03988478, -0.03000684,  0.01141376])
```

Som det næste vil vi definere en funktion, `distance`, som udregner afstanden mellem to punkter/vektorer. Denne bruger funktionen `numpy.linalg.norm(v)`, der udregner $\sqrt{v^2}$. Navnet `linalg` står for "linær algebra", som er den del af matematikken, der blandt andet handler om at regne med vektorer.

```
In [6]: def distance(a, b):
            return numpy.linalg.norm(a - b)
```

Vi afprøver derefter funktionen ved at finde afstanden mellem $(1, 2)$ og $(4, 6)$. Dette skal være 5, ligesom vi udregnede tidligere. Her bruger vi `numpy.array`, som laver en liste af tal om til en vektor, så computeren kan regne på den.

```
In [7]: distance(numpy.array([1, 2]), numpy.array([4, 6]))

Out[7]: 5.0
```

Afstanden mellem "Peter Plys" og "Frøken Smilla" er:

```
In [8]: distance(bibdata.genres[peter_plys], bibdata.genres[smilla])

Out[8]: 1.3939129763199098
```

Afstanden mellem "Cirkeline" og "Cykelmyggen Egon" er:

```
In [22]: distance(bibdata.genres[cirkeline], bibdata.genres[cykelmyggen])

Out[22]: 1.0181284635824785
```

Afstanden mellem rejsebøger om "Berlin" og "Paris" er:

```
In [23]: distance(bibdata.genres[berlin], bibdata.genres[paris])

Out[23]: 0.25173324294787786
```

Afstanden mellem rejsebog om "Berlin" og en håndarbejdsbog er:

```
In [24]: distance(bibdata.genres[berlin], bibdata.genres[strikning])
```

```
Out[24]: 1.3969369158183382
```

Afstanden mellem to bøger af "Haruki Murakami" er:

```
In [25]: distance(bibdata.genres[q84], bibdata.genres[kafka])
```

```
Out[25]: 0.4369157294393759
```

Konklusionen er, at afstanden mellem bøger i genrerummet faktisk giver mening. De tre forventninger, som jeg formulerede i starten af kapitlet, holder.

Bemærk at forventningerne blev formuleret, før eksperimenterne blev programmeret og kørt. Når man laver data science / videnskabelige eksperimenter, gælder det om først at formulere hypotese, og hvorledes man kan teste den, - og derefter, at udføre testen for, at se om hypotesen faktisk holder.

6 Case: Anbefalinger

TODO

- Work through text

```
In [1]: import bibdata
        import numpy

In [14]: [peter_plys, smilla, cirkeline, cykelmyggen, berlin, paris, strikning,
            q84, kafka] = [8944, 8214, 616, 580, 149, 278, 126, 29, 688]

In [3]: def distance(a, b):
            return numpy.linalg.norm(a - b)
```

Genrerummet bør også kunne bruges til, at finde litterære anbefalinger.

Lad os vælge en bog og derefter finde afstanden fra denne til alle andre bøger. Vi inkluderer titel/forfatter for at gøre resultatet mere læsbart. Når vi skriver `[:10]` betyder det, at vi kun viser de første 10 resultater i stedet for hele listen.

```
In [4]: distances_from_peter_plys = [
            (distance(bibdata.genres[peter_plys], bibdata.genres[other_book]),
                bibdata.title_creator(other_book))
            for other_book in range(0, 10000)
        ]
        distances_from_peter_plys[:10]

Out[4]: [(1.418060160516152, 'Fifty shades - E. L. James (book)'),
         (1.4149314310461802, 'Journal 64 : krimithriller - Jussi Adler-Olsen (book)'),
         (1.4123153810862557,
          'Marco effekten : krimithriller - Jussi Adler-Olsen (book)'),
         (1.4101266227823486, 'Taynikma - Jan Kjær (f. 1971) (book)'),
         (1.4100973534281041, 'De glemte piger : krimi - Sara Blædel (book)'),
         (1.4094550396442684,
          'Den grænseløse : krimithriller - Jussi Adler-Olsen (book)'),
         (1.4196622881449064, 'Vildheks - Lene Kaaberbøl (book)'),
         (1.4090062090707516, 'Dødesporet : krimi - Sara Blædel (audiobook)'),
         (1.4105623099966649, 'Dødsenglen : krimi - Sara Blædel (book)'),
         (1.4142121345574907,
          'Flaskepost fra P : krimithriller - Jussi Adler-Olsen (book)')]
```

Hvis vi nu sorterer listen af bøger efter afstanden til den valgte bog, så får vi en liste af anbefalinger.

```
In [5]: sorted(distances_from_peter_plys)[:10]
```

```
Out[5]: [(0.0, 'Batman - Arkham origins - WB Games Montréal (game)'),
         (0.10101559296838118, "Assassin's creed IV - black flag - Ubisoft (game)"),
         (0.10104348267424638, 'Diablo III -  (game)'),
         (0.11745171011979552, 'Grand theft Auto 5 - Rockstar North (game)'),
         (0.12725059983181469, 'Call of duty - Ghosts -  (game)'),
         (0.13485244625302031, 'The walking dead - Kirkman Robert (game)'),
         (0.13888979693702266, 'Far cry 3 - Crytek (game)'),
         (0.13991238369133902, 'Battlefield 4 - EA Digital Illusions (game)'),
         (0.14070083903851371,
          'Injustice - gods among us - Netherrealm Studios (game)'),
         (0.14073733392685148, 'The last of us - Naughty Dog (firma) (game)')]
```

Vi kan nu definere dette i en funktion, hvor vi kun returnerer titlerne. Hvis man i programmering har et par data: `rec = (afstand, titel)` kan man få fat i `titel` ved at skrive `rec[1]` (og få fat i `afstand` ved at skrive `rec[0]`).

```
In [6]: def recommendations(book):
            return [recommendation[1] for recommendation in
                sorted([
                    (distance(bibdata.genres[book], bibdata.genres[other_book]),
                        bibdata.title_creator(other_book))
                    for other_book in range(0, 10000)])]
```

Med denne funktion kan vi så udforske anbefalingerne til forskellige bøger:

```
In [7]: recommendations(smilla)[:10]
```

```
Out[7]: ['Frøken Smillas fornemmelse for sne : roman - Peter Høeg (book)',
         'Den kroniske uskyld - Klaus Rifbjerg (audiobook)',
         'Vinter-Eventyr - Karen Blixen (book)',
         'Kongens Fald - Johannes V. Jensen (f. 1873) (book)',
         'Ved Vejen - Herman Bang (book)',
         'Rend mig i traditionerne - Leif Panduro (audiobook)',
         'Kronprinsessen : roman - Hanne-Vibeke Holst (book)',
         'Det forsømte forår - Hans Scherfig (book)',
         'Drageløberen - Khaled Hosseini (book)',
         'Pelle Erobreren : barndom - Martin Andersen Nexø (book)']
```

```
In [8]: recommendations(cirkeline)[:10]
```

```
Out[8]: ['Cirkeline bliver til - Hanne Hastrup (book)',
         'Godmorgen Cirkeline - Hanne Hastrup (book)',
         'Cirkeline - tæl til 10 - Hanne Hastrup (book)',
         'Godnat Cirkeline - Hanne Hastrup (book)',
         'Cirkeline på opdagelse - Ulla Raben (book)',
         'Cirkeline godnat - Hanne Hastrup (book)',
         'Den store bog om Cirkeline - Hanne Hastrup (book)',
         'Cirkeline flytter til byen - Hanne Hastrup (book)',
         'Kender du Pippi Langstrømpe? : billedbog - Ingrid Vang Nyman (book)',
         'Cirkeline - Hanne Hastrup (book)']
```

```
In [9]: recommendations(cykelmyggen)[:10]

Out[9]: ['Cykelmyggen Egon - Flemming Quist Møller (book)',
         'Den store bog om den glade løve - Louise Fatio (book)',
         'Bennys badekar - Flemming Quist Møller (book)',
         'Pandekagekagen - Sven Nordqvist (book)',
         'Stakkels Peddersen - Sven Nordqvist (book)',
         'Gok-gok i køkkenhaven - Sven Nordqvist (book)',
         'Findus flytter hjemmefra - Sven Nordqvist (book)',
         'Og hanen gol - Sven Nordqvist (book)',
         'Paddington - Michael Bond (book)',
         'Rasmus får besøg - Jørgen Clevin (book)']

In [10]: recommendations(berlin)[:10]

Out[10]: ['Turen går til Berlin - Therkelsen Kirstine (book)',
          'Politikens visuelle guide - Berlin - Søndervang Allan edt (book)',
          'Top 10 Berlin - Jürgen Scheunemann (book)',
          'Turen går til Amsterdam - Anette Jorsal (book)',
          'Turen går til Prag - Hans Kragh-Jacobsen (book)',
          'Turen går til Hamburg og Nordtyskland - Jytte Flamsholt Christensen (book)',
          'Turen går til Californien & det vestlige USA - Preben Hansen (f. 1956) (book)',
          'Turen går til London - Gunhild Riske (book)',
          'Politikens Kort og godt om Berlin - Charmetant Jim (book)',
          'Politikens visuelle guide - Prag - Vladimír Soukup (book)']

In [11]: recommendations(strikning)[:10]

Out[11]: ['Alt om håndarbejdes strikkemagasin -  (other)',
          'Kreativ strik -  (other)',
          "Ingelise's strikkemagasin -  (other)",
          'DROPS : strikkdesign - Garnstudio (periodica)',
          'Alt om håndarbejdes symagasin -  (other)',
          'Maries ideer : håndarbejde, strik, sy, hækl, bolig, inspiration -  (other)',
          'Burda style -  (other)',
          'Burda -  (other)',
          'Burda modemagasin : verdensberømt mode -  (other)',
          "Ingelise's symagasin -  (other)"]

In [12]: recommendations(kafka)[:10]

Out[12]: ['Kafka på stranden - Haruki Murakami (book)',
          'Trækopfuglens krønike - Haruki Murakami (book)',
          'Norwegian wood - Haruki Murakami (book)',
          'Sønden for grænsen og vesten for solen - Haruki Murakami (book)',
          'Efter midnat - Haruki Murakami (book)',
          'Sputnik min elskede - Haruki Murakami (book)',
          'Hvad jeg taler om når jeg taler om at løbe - Haruki Murakami (book)',
          'Brooklyn dårskab : roman - Paul Auster (book)',
          '1Q84 - Haruki Murakami (audiobook)',
          'Efter skælvet - Haruki Murakami (book)']
```

Vi har hermed lavet koden for en lille anbefalingsservice.

For perspektivering, sammenlign eksempelvis resultaterne med anbefalingerne på bibliotekernes hjemmesider. Bemærk at vi her kun kigger på et lille udvalg af materialebestanden. På bibliotek.dk kan anbefalinger findes ved, at fremsøge en bog, og derefter klikke på "Inspiration", og "Andre der har lånt...". Genrerummet som vi benytter her, er faktisk udregnet fra (en lille delmængde af) de "Andre der har lånt"-data, som DBC havde med på Hack4DK.

Bemærk at bøgerne, som vi finder anbefalinger for, er valgt på forhånd (og derved uafhængigt af hvordan anbefalingerne bliver). Derfor må kvaliteten af anbefalinger forventes at være repræsentativ.

6.1 Øvelser

- Ændr antallet af resultater der vises for nogle af de ovenstående bøger.
- Udforsk anbefalingerne for andre bøger end de ovenstående.

7 Case: Emneord

TODO - text

```
In [1]: import bibdata
        from collections import Counter
        import numpy

In [2]: bibdata.meta[1234]

Out[2]: {'_id': '870970-basis:51363035',
         'creator': 'Jakob Martin Strid',
         'idx': 1234,
         'language': 'Dansk',
         'subject-term': ['årstider',
          'for 5 år',
          'sk',
          'sjove bøger',
          'Skønlitteratur',
          'for 4 år',
          'for 6 år',
          'for 3 år',
          'venner',
          'vejret',
          'dyrefortællinger'],
         'title': 'Mimbo Jimbo og den lange vinter',
         'type': 'book'}

In [3]: subjects = []
        for bib in bibdata.meta:
            for subject in bib.get('subject-term', []):
                subjects.append(subject)
        Counter(subjects).most_common(10)

Out[3]: [('sk', 7036),
         ('Skønlitteratur', 6445),
         ('for 6 år', 1109),
         ('krimi', 1100),
         ('for 5 år', 1097),
         ('for 7 år', 1063),
```

```
         ('let at læse', 1047),
         ('for 4 år', 999),
         ('for 8 år', 932),
         ('Spillefilm', 904)]
```

```
In [4]: "; ".join([subject for (subject, count) in Counter(subjects).most_common(150)])
```

```
Out[4]: 'sk; Skønlitteratur; for 6 år; krimi; for 5 år; for 7 år; let at læse; for 4 år; for
```

Lad os udforske hvilke typer af biblioteksmaterialer der er:

```
In [5]: Counter([bib.get('type') for bib in bibdata.meta]).most_common(5)
```

```
Out[5]: [('book', 7895),
         ('movie', 1055),
         ('audiobook', 608),
         ('game', 232),
         ('other', 119)]
```

Gad vide hvad 'other' dækker over:

```
In [6]: [bib['title'] for bib in bibdata.meta if bib.get('type') == 'other'][:5]
```

```
Out[6]: ['Alt for damerne', 'Illustreret videnskab', 'Femina', 'Bo bedre', 'I form']
```

```
In [7]: def books_by_subject(subject):
            books = [book_id for book_id in range(len(bibdata.meta)) if subject in bibdata.me
            vectors = [bibdata.genres[book] for book in books]
            mean = numpy.mean(vectors, axis=0)
            std = numpy.std(vectors, axis=0)
            weighted_books = [(numpy.linalg.norm((bibdata.genres[book_id]-mean)/std), book_id
            sorted_books = [book_id for (weight, book_id) in sorted(weighted_books)]
            return sorted_books
```

```
In [8]: "; ".join([subject for (subject, count) in Counter(subjects).most_common(150)])
```

```
Out[8]: 'sk; Skønlitteratur; for 6 år; krimi; for 5 år; for 7 år; let at læse; for 4 år; for
```

```
In [9]: subject = 'Biografier af enkelte personer'
        subject = 'computerspil'
        print('\n'.join([bibdata.title_creator(book_id) for book_id in books_by_subject(subje
         if not subject in bibdata.meta[book_id].get('subject-term', [])][:5]))
        print('---- subject:' + subject + ' ----')
        print('\n'.join([bibdata.title_creator(book_id) for book_id in reversed(books_by_subj
         if subject in bibdata.meta[book_id].get('subject-term', [])][:5]))
```

```
Surf's up - Ubi Soft (game)
Spider-man 3 -  (game)
Asterix at the Olympic Games - René Goscinny (game)
Rayman - raving rabbids - Ubi Soft (game)
Brave : the search for Spirit Dancer - Mark Hughes (game)
---- subject:computerspil ----
Det 13. spil : roman - Morten Sabroe (book)
Ripper : krimi - Isabel Allende (book)
```

```
Tron - legacy - Walt Disney firma (movie)
De ukendte - Jan Solheim (book)
Minecraft : byggehåndbogen - Mojang (book)
```

```
In [10]: bibdata.meta[9953]
```

```
Out[10]: {'_id': '870970-basis:50673103',
          'creator': 'Dowsett Elizabeth edt',
          'idx': 9953,
          'language': 'Dansk',
          'subject-term': ['!',
           'for 12 år',
           'for 11 år',
           'Biografier af enkelte personer',
           'for 9 år',
           'Dave Filoni',
           'Star wars - the clone wars',
           '99.4 Filoni, Dave',
           'Legetøj',
           '79.31',
           'figurer',
           'Dave',
           'for 10 år',
           'Filoni, Dave',
           'Filoni'],
          'title': 'Star wars - clone wars figurleksikon',
          'type': 'book'}
```

```
In [11]: subject = 'let at læse'
         # books = [book_id for book_id in range(len(bibdata.meta)) if subject in bibdata.met
         books = [book_id for book_id in range(len(bibdata.meta)) if subject in bibdata.meta[
         vectors = [bibdata.genres[book] for book in books]
         mean = numpy.mean(vectors, axis=0)
         std = numpy.std(vectors, axis=0)

         subject_books = sorted(
             [(numpy.linalg.norm((bibdata.genres[book_id]-mean)/std, ord=1), book_id) for boo
         [(bibdata.title_creator(book_id), ",".join(bibdata.meta[book_id].get('subject-term',
```

```
Out[11]: [('Jorden - Leonie Pratt (book)',
           'Anbefales: Basis,for 9 år,Planeter. Måner,for 8 år,÷,for 7 år,Jorden,52.43,Lette
          ('Rumfart - Troels Gollander (book)',
           'Rummet,Første Fakta,Sagprosa sm,for 9 år,Faglig læsning,Tværfagligt materiale,for
          ('Kom frem, Bella! - Otto Dickmeiss (book)',
           'for 6-7 år,bange,sk,Skønlitteratur,for 7 år,for 6 år,hunde'),
          ('Aladdin og den magiske lampe - Katie Daynes (book)',
           'for børn,Iran,39.12,eventyr,for 9 år,iranske eventyr,sk,for 8 år,Skønlitteratur,r
          ('Alle elsker Sigge - Benedikte Biørn (audiobook)',
           'heste,for 9 år,veninder,sk,for 8 år,Skønlitteratur,for 10 år,piger'),
          ('Pony & co. - Kirsten Sonne Harild (audiobook)',
           'for 11 år,heste,for 9 år,veninder,sk,Skønlitteratur,for 10 år,piger'),
```

```
('Gode venner og et straffespark - Christian Bieniek (book)',
 'fodbold,for 12 år,for 11 år,SerieM1,for 9 år,sk,Skønlitteratur,venskab,for 10 år,
('Torsdage med Sigge - Benedikte Biørn (audiobook)',
 'heste,for 9 år,veninder,sk,for 8 år,Skønlitteratur,for 10 år,piger'),
('Teddy til salg - Margareta Nordqvist (book)',
 'for højtlæsning,heste,for 9 år,sk,for 8 år,Skønlitteratur,for 7 år,for 6 år,piger
('Firkløveret - Christina Nordstrøm (book)',
 'Ridning,for 11 år,for 9-11 år,heste,for 9 år,pigeliv,sk,Heste,Skønlitteratur,for
```

8 Case: Genre-klynger

TODO

- rewrite/read through

Jeg er nysgerrig om hvilke typer / kategorier af biblioteksmaterialer, der er. Særligt i litteratur, som ligger ud over klassifikationssystemer som DK5. Derfor vil jeg lave en mere udforskende / eksplorativ analyse. En tilgang til detter er, at bede computeren, at sortere biblioteksmaterialerne i et antal bunker(klynger), hvor ting, der ligner hinanden, ligger i samme bunke(klynge).

Dette kaldes klynge-analyse, og en af de enkleste måder at gøre dette kaldes k-means. Her starter man med at fortælle hvor mange bunker man ønsker. Dette antal kaldes ofter k. Algoritmen virker således: vi har et punkt i genrerummet for hver bunke. Hver bog lægges i den bunke, hvor afstanden mellem bunkens punkt og bogen er kortest. Derefter flyttes punktet for hver bunke, til gennemsnits-punktet for alle bøgerne i bunken. Dette gentager man indtil, at punkterne for bunkerne ikke længere flytter sig.

Bemærk at vi her beder computeren om, selv at finde en struktur i data. Dette kaldes unsupervised learning indenfor machine learning / data mining.

Når vi skal programmere det, findes k-means algoritmen allerede i Python's SciKit-Learn funktionsbibliotek. Så vi behøver blot importere det, fortælle at vi ønsker k-means med k klynger, samt at den skal køre algoritmen på genrerummet:

```
In [6]: import bibdata
        import sklearn.cluster
        k = 15
        kmeans = sklearn.cluster.KMeans(n_clusters = k)
        kmeans.fit(bibdata.genres)

Out[6]: KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
            n_clusters=15, n_init=10, n_jobs=1, precompute_distances='auto',
            random_state=None, tol=0.0001, verbose=0)
```

Nu har computeren fundet nogle klynger, som vi kan kigge på.

Vi vil nu se hvad klyngerne indeholder. For hver klynge udskriver vi derfor titel/forfatter for de første bøger i hver klynge. Koden for dette læses lettest bagfra: `cluster_id` er nummeret på klyngen. For hvert `cluster_id` gennemløber alle 10000 bøger, og udvælger dem som faktisk ligger i klyngen. Til dette bruger vi `kmeans.labels_`, som er en liste over hvilken klynge hver bog hører til. Herefter udvælger vi de første `book_count` bøger, som vi returnere, så vi faktisk kan se hvad der er i klyngen:

In [7]:
```
book_count = 7
[[bibdata.title_creator(book_id)
  for book_id in range(10000)
  if kmeans.labels_[book_id] == cluster_id][0:book_count]
 for cluster_id in range(k)]
```

Out[7]:
```
[['Os fra Blomsterkvarteret - Dy Plambeck (book)',
  'Troldmandens flyvefedt - Bent T. Lund (f. 1946) (book)',
  'Klods-Hans - H. C. Andersen (f. 1805) (audiobook)',
  'Hvad kan det være? - Keld Petersen (f. 1955) (book)',
  'Kejserens nye klæder - Birte Dietz (book)',
  'Den grimme ælling - Svend Otto (book)',
  'Læs og gæt gåder - Keld Petersen (f. 1955) (book)'],
 ['Naboens søn : roman - Jane AAmund (book)',
  'Knud, den store : roman - Hanne-Vibeke Holst (book)',
  'Zornig - vrede er mit mellemnavn - Lisbeth Zornig Andersen (f. 1968) (book)',
  'Vindue uden udsigt - Jane AAmund (book)',
  'Vi kan sove i flyvemaskinen - Ulla Terkelsen (audiobook)',
  'Hvor solen græder : en fortælling fra Syrien - Puk Damsgård Andersen (book)',
  'Onklerne og deres fruer : erindringsroman (mp3) - Jane Aamund (audiobook)'],
 ['Fifty shades - E. L. James (book)',
  'Journal 64 : krimithriller - Jussi Adler-Olsen (book)',
  'Marco effekten : krimithriller - Jussi Adler-Olsen (book)',
  'Taynikma - Jan Kjær (f. 1971) (book)',
  'De glemte piger : krimi - Sara Blædel (book)',
  'Den grænseløse : krimithriller - Jussi Adler-Olsen (book)',
  'Dødesporet : krimi - Sara Blædel (audiobook)'],
 ['Vildheks - Lene Kaaberbøl (book)',
  'Harry Potter og De Vises Sten - Joanne K. Rowling (book)',
  'Skammerens datter - Lene Kaaberbøl (book)',
  'Harry Potter og Hemmelighedernes Kammer - Joanne K. Rowling (book)',
  'Mustafas kiosk - Jakob Martin Strid (book)',
  'Wimpy Kid - Jeff Kinney (book)',
  'Gummi-Tarzan - Ole Lund Kirkegaard (audiobook)'],
 ['Vi unge : kun for piger -  (other)',
  'Livs liv - Rikke Dyrhave Nissen (book)',
  'Pretty little liars - Sara Shepard (book)',
  'Knus-klubben - Gitte Løkkegaard (book)',
  'Lego Indiana Jones - the original adventures - Glyn Scragg (game)',
  'Olivia -  (other)',
  'Toy story 3 - Walt Disney (firma) (game)'],
 ['Stå fast : et opgør med tidens udviklingstvang - Svend Brinkmann (book)',
  'Spis maven flad : sådan gør du! : 111 skønne opskrifter - Engell Majbritt L. aut (
  'Tarme med charme : alt om et undervurderet organ - Giulia Enders (book)',
  'Hvad kan jeg blive? : Politikens erhvervsvejledning - Dueled Knud (book)',
  'Verdens bedste kur : vægttab der holder - Bitz Christian aut (book)',
  'Dr. Zukaroffs testamente : en bog om menneskehjernen - Peter Lund Madsen (book)',
  'Stenalderkost : palæo-opskrifter til det moderne menneske - Thomas Rode Andersen (
 ['Min mormors gebis - Jakob Martin Strid (book)',
  'Kender du Pippi Langstrømpe? : billedbog - Ingrid Vang Nyman (book)',
  'Pippi Langstrømpe på Kurrekurredut-øen - Ingrid Vang Nyman (book)',
  'Lille frø - Jakob Martin Strid (book)',
```

```
 'Da Findus var lille og forsvandt - Sven Nordqvist (book)',
 'Fyrtøjet - H. C. Andersen (f. 1805) (book)',
 'Da Lille Madsens hus blæste væk - Jakob Martin Strid (book)'],
['Turen går til Hamburg og Nordtyskland - Jytte Flamsholt Christensen (book)',
 'Turen går til London - Gunhild Riske (book)',
 'Turen går til Rom - Alfredo Tesio (book)',
 'Turen går til Mallorca, Menorca & Ibiza - Jytte Flamsholt Christensen (book)',
 'Turen går til Tyrkiet - Fenger-Grøndahl Carsten (book)',
 'Turen går til Berlin - Therkelsen Kirstine (book)',
 'Turen går til Sydspanien - Jørgen Laurvig (book)'],
['Kriminalkommissær Barnaby - Caroline Graham (movie)',
 'Avatar - Fiore Mauro cng (movie)',
 'Hævnen - Bier Susanne (movie)',
 'Dirch - Zandvliet Martin Pieter (movie)',
 'The godfather - Willis Gordon (movie)',
 'Flammen & Citronen - Andersen Lars K. f. 1960-11-27 (movie)',
 'Black swan (blu-ray) - maclaughlin McLaughlin John J. (movie)'],
['Bamses billedbog dvd - Katrine Hauch-Fausbøll (movie)',
 'Kaj og Andrea - Fausbøll Katrine Hauch- (movie)',
 'Ninjago - masters of spinjitzu - Sørensen Thomas inv (movie)',
 'Pippi - Hellbom Olle (movie)',
 'Filmhits for børn -  (movie)',
 'Legends of Chima - Andreasen Tommy inv (movie)',
 'Op. Med bonus features disc - Docter Pete (movie)'],
['Niceville - Kathryn Stockett (book)',
 'Jeg skal gøre dig så lykkelig - Anne B. Ragde (audiobook)',
 'Den stjålne vej - Anne-Cathrine Riebnitzsky (book)',
 'Øen - Victoria Hislop (book)',
 'Jordemoderen fra Hope River - Patricia Harman (audiobook)',
 'Fra hus til hus : roman - Kristín Marja Baldursdóttir (book)',
 'Orkideens hemmelighed - Lucinda Riley (audiobook)'],
['Den gode opgave : opgaveskrivning på videregående uddannelser - Rienecker Lotte (b
 'Interview : en introduktion til det kvalitative forskningsinterview - Steinar Kval
 'Læring : aktuel læringsteori i spændingsfeltet mellem Piaget, Freud og Marx - Knud
 'Kvalitative metoder : en grundbog - Tanggaard Lene edt (book)',
 'Klassisk og moderne samfundsteori - Kaspersen Lars Bo edt (book)',
 'Helbredets mysterium : at tåle stress og forblive rask - Aaron Antonovsky (book)',
 'Modernitet og selvidentitet : selvet og samfundet under sen-moderniteten - Anthony
['Alt om håndarbejdes symagasin -  (other)',
 'Alt om håndarbejdes strikkemagasin -  (other)',
 'Alt om haven -  (other)',
 'Haven - Det Jyske Haveselskab (other)',
 'Danmarks store gør det selv leksikon - Nielsen Jørn (book)',
 'Burda -  (other)',
 'Politikens strikke- og hæklebog - Vivian Høxbro (book)'],
['Giganternes fald - Ken Follett (audiobook)',
 'Og bjergene gav genlyd - Khaled Hosseini (book)',
 'Det syvende barn - Erik Valeur (audiobook)',
 'Den hundredårige der kravlede ud ad vinduet og forsvandt - Jonas Jonasson (book)',
 '1Q84 - Haruki Murakami (audiobook)',
 'Min kamp : roman - Karl Ove Knausgård (book)',
```

```
 'Brobyggerne - Jan Guillou (book)'],
['Det blinde punkt : krimi - Julie Hastrup (book)',
 'Ensomme hjerters klub : kriminalroman - Lotte Hammer (book)',
 'Skrig under vand - Ole Tornbjerg (book)',
 'Alting har sin pris : kriminalroman - Lotte Hammer (book)',
 'Skindød - Thomas Enger (book)',
 'Møgkællinger - Gretelise Holm (f. 1946) (book)',
 'Manden der ikke var morder - Hans Rosenfeldt (book)']]
```

Hvis vi kører klyngeanalysen igen, vil den komme med nogle andre klynger, men strukturen i resultatet vil være det samme, i.e. rejsebøger, børnefilm, forskellige typer krimier, forskellige typer børnebøger, andre forskellige typer romaner, studiebøger, etc.

Resultatet er i mine øjne interessant. Blandt andet ser jeg forskellige separate typer af skønlitteratur, hvis opdeling jeg ikke kendte til i forvejen. Det kunne være interessant, at køre dette mere finkornet, på større datasæt, og derefter danne nogle foreslag til emnehierakier for skønlitteratur, for at få mere indblik i denne del af litteraturen.

8.1 Øvelser

- Få computeren til at udføre denne notebook flere gange, og se hvordan de fundne klynger ændre sig, men stadigt har noget af den samme struktur. (Rækkefølgen ændres også, så læg i stedet mærke til karakteristiske klynge, og se at/om de går igen).
- Ændr antallet, `k`, af fundne klynger. Prøv eksempelvis med 25 eller flere, hvorved resultatet bliver mere finkornet.
- Ændr det maksimale antal af bøger `book_count`, som bliver vist fra hver klynge. Hvis man viser mere end de nuværende 7 materialer, giver det mere indblik i klyngen (men ville også bruge mere papir/skærmplads).

9 Efterskrift

TODO

- Konklusion
- Links til videre studier