



NTNU

Norwegian University of Science and Technology

Dictionary learning for classification problems

TMA4320 Prosjekt 2 Vår 2023

Martin Ludvigsen

Institutt for Matematiske fag, NTNU.

Praktisk informasjon

- ▶ Prosjektet teller 15 % av karakteren, poengskala 0-10.
- ▶ Jobb i grupper på 3.
- ▶ Innleveringsfrist er 6. Mars.
- ▶ Oppgaveteksten er på engelsk, men lever på norsk om dere vil.
- ▶ Lever en jupyter notebook i .ipynb format og sørg for at dere har kjørt all kode før dere leverer!
- ▶ Bruk veiledningstimer (se wiki-siden) og diskusjonsforumet Mattelab godt!
- ▶ Dere får utlevert en .ipynb fil som lar dere laste inn data og som viser hvordan dere kommer i gang.
- ▶ Prosjektet har mye rom for eksperimentering, men sørg for at dere **svarer tydelig på oppgaveteksten.**
- ▶ Prosjektet er langt og vanskelig, fokuser på å gjennomfør så mye dere kan! De første oppgavene er enklere og teller mer.
- ▶ Relativt lite implementering, mer teori og forståelse.
- ▶ Skriv rapport underveis!

Introduksjon

Noen tiår siden: ingeniørvirksomhet, forskning og kommersiell virksomhet er for det meste **kunnskapsbasert**.

I dag: flere og flere løsninger er **datadrevet**. AI og Maskinlæring er mer effektiv enn menneskelige eksperter.

Eksempel: Klassifisering av kreft

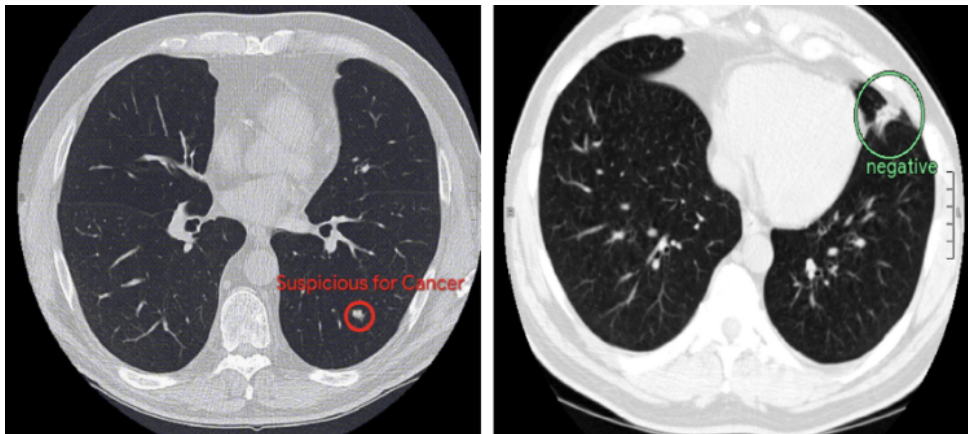


Figure: AI kan klassifisere kreft bedre enn trenete leger. Kilde:
<https://venturebeat.com/ai/googles-lung-cancer-detection-ai-outperforms-6-human-radiologists/>

Eksempel: Generering av musikk-spillelister

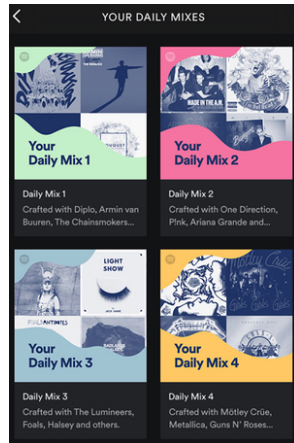


Figure: Mixtapes lagd av "eksperter" er nå AI-genererte Daily Mixes.

Eksempel: ChatGPT



Give me an example of a data-driven method that used to be knowledge-based in the past that would be suitable for second year students with little to no experience with machine learning.



One example of a data-driven method that used to be knowledge-based in the past is spam filtering. In the past, email providers used rule-based systems that were manually created to filter out spam emails. Nowadays, machine learning algorithms are widely used for spam filtering.



Data-drevne metoder

Mere tillitt til data-drevne metoder \implies

- ▶ Mere data-kolleksjon.
- ▶ Mere forskning.
- ▶ Mere beregningsressurser spesifikt for data-drevne metoder (GPU).
- ▶ Mere media-dekning.

... som igjen fører til mere tillitt til data-drevne metoder.

Problem: Så mye data at vi trenger metoder for å forstå og anvende data.

Akt 1: Dictionary learning

Dataseett som matriser

Dataseett kan lagres kollonnevis i en $m \times n$ matrise A .

- ▶ m , Størrelsen på data, f.eks antall piksler i et bilde.
- ▶ n , Antall data.

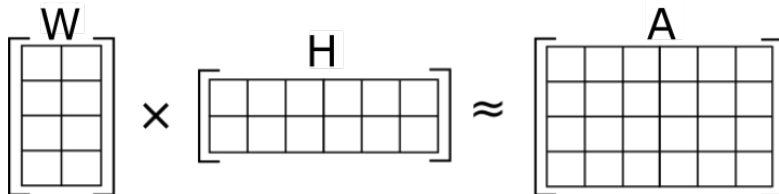
Hver kolonne av A , $a_i \in \mathbb{R}^m$, samsvarer med ett datapunkt, f.eks et bilde.

Dictionary learning og matrisefaktorisering

Dictionary learning: Trekk ut (lineære) basis-vektorer fra datasett i en matrise W .

En måte å gjøre dette på er å regne ut en **matrisefaktorisering** $A \approx WH$.

- ▶ $W \in \mathbb{R}^{m \times d}$, dictionary.
- ▶ $H \in \mathbb{R}^{d \times n}$, vektor/latente variabler.
- ▶ $d \ll m, n$, antall basisvektorer.



Dimensionality reduction

Hver kolonne av datasettet a_i kan da beskrives som en lineær-kombinasjon:

$$a_i \approx w_1 h_{i1} + w_2 h_{i2} + \dots \quad (1)$$

Hver basis-vektor w_j burde inneholde interessante **features** ved datasettet.

Dette kalles også **dimensionality reduction**.



Figure: Vi skal se på det kjente MNIST datasettet som består av 70000 håndskrevne tall.

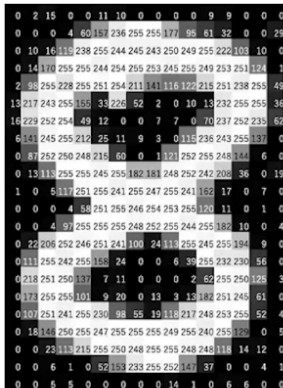


Figure: Et gråskala bilde hvor pikselverdiene er skrevet inn i hver piksel.

- ▶ $28 \times 28 = 784$ piksler.
- ▶ Gråskala bilder, hver pikselverdi er mellom 0 og 255, eller 0 og 1.
- ▶ Hvert bilde kan representeres som en \mathbb{R}^m vektor, med $m = 784$.
- ▶ Et datasett med n bilder kan representeres som en $m \times n$ matrise/NumPy array.
- ▶ Hver kolonne er ett bilde og hver rad er en piksel.
- ▶ Hvert bilde er sentrert og skalert, som gjør læring mye enklere.

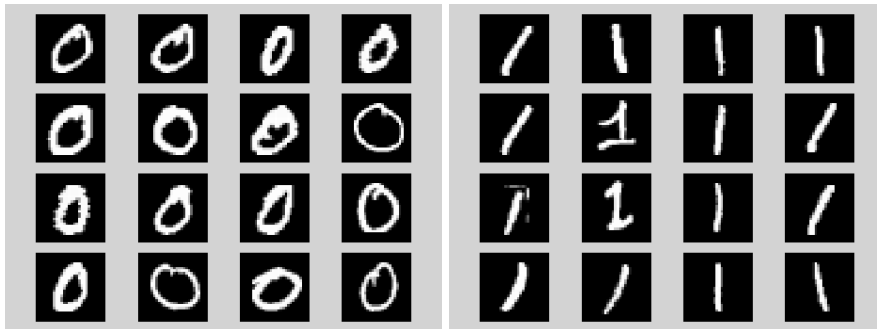


Figure: Hvordan ser en dictionary ut for forskjellige typer data? Hvilke interessante features inneholder et håndskrevet nulltall sammenlignet med et ettall?



Eigenverdifaktorisering

Fra Matte 3 vet vi at en $n \times n$ matrise A kan **diagonaliseres**:

$$A = PDP^{-1}, \quad (2)$$

hvor P inneholder egenvektorene til A som kolonner og D er diagonal med egenverdiene til A .

Hvis egenvektorene er ortonormal er P *ortogonal* ($P^T = P^{-1}$, $P^T P = PP^T = I$), og

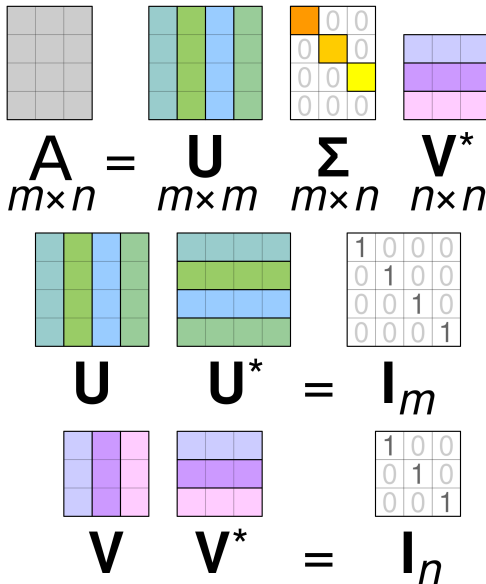
$$A = PDP^T. \quad (3)$$

Kan bruke dette til å lære dictionaries: $W = P$, $H = DP^T$.

Singular Value Decomposition (SVD)

Problem: Hva hvis A er rektangulær?

Løsning: Singulærverdi dekomposisjon (SVD)



The diagram illustrates the SVD decomposition of a matrix A into three matrices: U , Σ , and V^* .

Matrix A: A 4x4 grid of gray squares.

Matrix U: A 4x4 grid with columns colored teal, green, blue, and green.

Matrix Σ: A 4x4 grid with a diagonal of colored squares (orange, yellow, yellow, white) and zeros elsewhere.

Matrix V*: A 4x4 grid with rows colored light blue, purple, purple, and pink.

The equation is shown as:

$$A = U \Sigma V^*$$

Below the main equation, the dimensions of the matrices are specified:

$$m \times n \quad m \times m \quad m \times n \quad n \times n$$

Orthogonality of U and V:

Matrix U: A 4x4 grid with columns colored teal, green, blue, and green.

Matrix U*: A 4x4 grid with rows colored teal, green, blue, and green.

Equation: $U U^* = I_m$

Matrix V: A 4x4 grid with columns colored light blue, purple, purple, and pink.

Matrix V*: A 4x4 grid with rows colored light blue, purple, purple, and pink.

Equation: $V V^* = I_n$

Identity Matrices:

I_m (4x4):

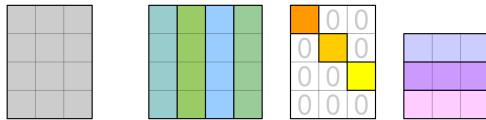
1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1

I_n (4x4):

1	0	0
0	1	0
0	0	1

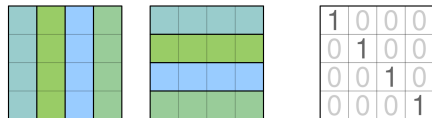
Singular Value Decomposition (SVD)

- ▶ U er ortogonal og inneholder venstre singulærvektorer
- ▶ V er ortogonal og inneholder høyre singulærvektorer.
- ▶ Σ inneholder singulærverdier, fra høyest til lavest.

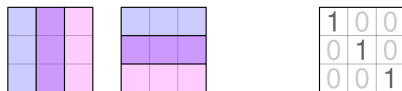


$$A = U \Sigma V^*$$

$m \times n$ $m \times m$ $m \times n$ $n \times n$



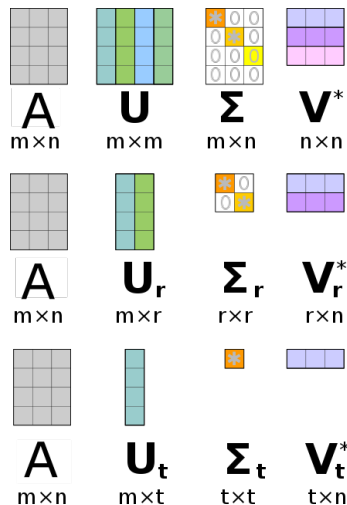
$$U U^* = I_m$$



$$V V^* = I_n$$

Singular Value Decomposition (SVD)

- Fjern singulærverdier og vektorer som samsvarer med singulærverdi lik 0 → **reduced SVD**, perfekt rekonstruksjon av A .
- Fjern singulærverdier og vektorer som samsvarer med lave singulærverdier → **truncated SVD**, beste lav-rang rekonstruksjon av A .



Ortogonale dictionaries fra SVD

1. Lagre datasett kolonnevis i en matrise $A \in \mathbb{R}^{m \times n}$.

Ortogonale dictionaries fra SVD

1. Lagre datasett kolonnevis i en matrise $A \in \mathbb{R}^{m \times n}$.
2. Beregn de første d vektorene til SVD-en av A , slik at $A \approx U_d \Sigma_d V_d^T$.

Ortogonale dictionaries fra SVD

1. Lagre datasett kolonnevis i en matrise $A \in \mathbb{R}^{m \times n}$.
2. Beregn de første d vektorene til SVD-en av A , slik at $A \approx U_d \Sigma_d V_d^T$.
 - ▶ ELLER: Beregn hele SVD-en (`np.linalg.svd`) og "klipp" bort unødvendige vektorer.
 - ▶ Beregning av SVD er **dyrt**, så dette fungerer kun for små datasett.

Ortogonale dictionaries fra SVD

1. Lagre datasett kolonnevis i en matrise $A \in \mathbb{R}^{m \times n}$.
2. Beregn de første d vektorene til SVD-en av A , slik at $A \approx U_d \Sigma_d V_d^T$.
 - ▶ ELLER: Beregn hele SVD-en (`np.linalg.svd`) og "klipp" bort unødvendige vektorer.
 - ▶ Beregning av SVD er **dyrt**, så dette fungerer kun for små datasett.
3. Ortogonal dictionary: $W = U_d \in \mathbb{R}^{m \times d}$ og vektor $H = \Sigma_d V_d^T \in \mathbb{R}^{d \times n}$.

Ortogonal: $W^T W = I_d$, men $W W^T \neq I_m$.

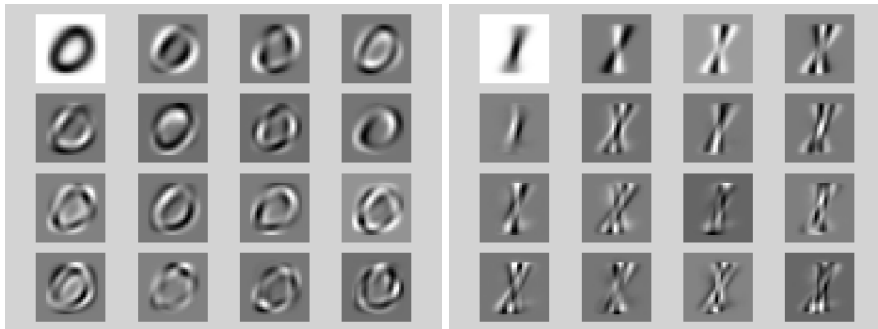


Figure: Denne framgangsmåten gir dictionaries med basisvektorer som ser slik ut. De forskjellige basisene fanger opp de viktigste egenskapene med dataene (?)

Trening vs. testing

- ▶ **Trening:** Lære strukturer fra data.
- ▶ **Testing:** Bruke lærte strukturer på nye data.
- ▶ Gitt en dictionary W , og vi observerer et nytt datapunkt b , kan vi finne h^* slik at $b \approx Wh^*$?
- ▶ (Har dere lært noe i Matte 3 om dette?)

Akt 2: Projesjoner

Projeksjon

Løsning: Finn h^* slik at Wh^* er så nærme b som mulig:

$$h^* = \arg \min_{h \in \mathbb{R}^d} \|b - Wh\|_2. \quad (4)$$

Minste kvadraters metode.

Analytisk løsning for ortogonal W :

$$h^* = W^T b. \quad (5)$$

Definer projeksjon:

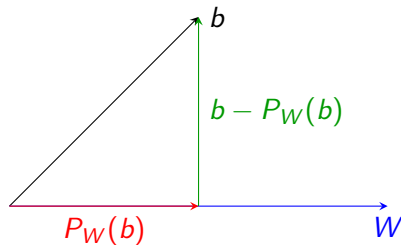
$$P_W(b) = WW^T b. \quad (6)$$

Ortogonal projeksjon

Definer avstand:

$$\begin{aligned} D_W(b) &= \|b - P_W(b)\|_2 \\ &= \min_{h \in \mathbb{R}^d} \|b - Wh\|_2 \end{aligned}$$

Kvantitativt mål på hvor nærme et datapunkt er dataen W er trent på.



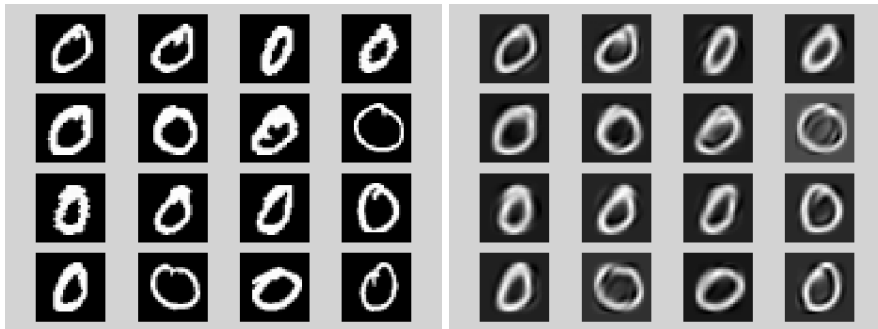


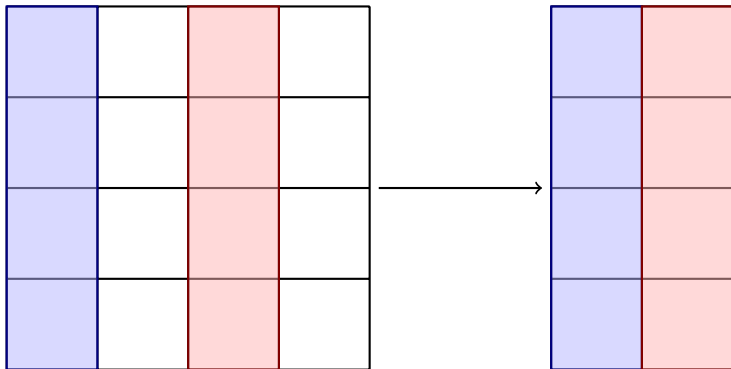
Figure: A og $P_W(A)$ hvor W er trent på A med $d = 32$ og $n = 5000$.

Før vi går videre...

Spørsmål?

Exemplar-based Non-Negative Matrix Factorization

Velg basisvektorer i W ved å trekke vektorer fra data A



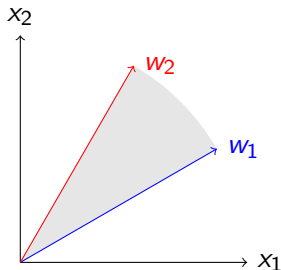
Data matrise A

Dictionary matrise W_+

- ▶ Dictionary er **ikke-negativ** når data er ikke-negativ, dvs alle innlegg er større eller lik null.
- ▶ Data kan deretter beskrives som $A \approx W_+ H_+$, hvor H_+ er ikke-negativ.
- ▶ Ikke-negativ lineær kombinasjon
- ▶ **Exemplar-based Non-Negative Matrix Factorization (ENMF).**
- ▶ (Tilnærmet) null trening! Hva med testing?

Ikke-negativ lineær algebra

Lineært uavhengige ikke-negative vektorer spenner ut det som kalles en **kjegle**, ikke et lineært underrom.



Testing: Projeksjon ned på en kjegle i steden for et lineært underrom.

Ikke-negative projeksjon

Vil finne H_+^* slik at $A \approx W_+ H_+^*$:

$$H_+^* = \arg \min_{H_+ \in \mathbb{R}_+^{d \times n}} \|A - W_+ H_+\|_F^2. \quad (7)$$

Ikke-negativ minste kvadraters metode.

Frobenius norm siden vi her projekterer en hel matrise i steden for en vektor.

Ingen analytisk løsning! Må bruke en numerisk metode...

Numerisk algoritme for ikke-negativ projeksjon

$$H_{k+1} \leftarrow H_k \odot (W^T A) \oslash (W^T W H_k + \delta), \quad (8)$$

hvor δ er en safe division/sparsity parameter (norsk: glissenhetsparameter).

\odot og \oslash er Hadamard produkt/divisjon

$$A \odot B = \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix} \odot \begin{bmatrix} b_1 & b_2 \\ b_3 & b_4 \end{bmatrix} = \begin{bmatrix} a_1 b_1 & a_2 b_2 \\ a_3 b_3 & a_4 b_4 \end{bmatrix}, \quad A \oslash B = \begin{bmatrix} a_1/b_1 & a_2/b_2 \\ a_3/b_3 & a_4/b_4 \end{bmatrix}. \quad (9)$$

Enkel å implementere, konvergerer på ~ 50 iterasjoner.

Ikke-negativ projeksjon

Projeksjonen på en kjegle spent av vektorer i W er

$$P_W^+(A) = W_+ H_+^*, \quad (10)$$

hvor H_+^* er beregnet med iterativ algoritme.

Avstand:

$$D_W^+(b) = \|b - P_{W_+}^+(b)\|_2. \quad (11)$$

Oppgave 1 og 2

- ▶ Implementere kode for å trene truncated SVD og ENMF.
- ▶ Implementere kode for projeksjon og avstand til ortogonale dictionaries og ikke-negative dictionaries.
- ▶ Teste dette på noen små eksempel-matriser (rundt 35 % av karakteren)
- ▶ Teste dette på MNIST data, og reflektere litt rundt resultatene (rundt 35 % av karakteren).
- ▶ Teste projeksjon ned på "feil" basis → relevant for oppgave 3.

Akt 3: Klassifisering

Klassifisering

Klassifisering handler om å lære datamaskiner til å predikere hvilken klasse et datapunkt hører til.

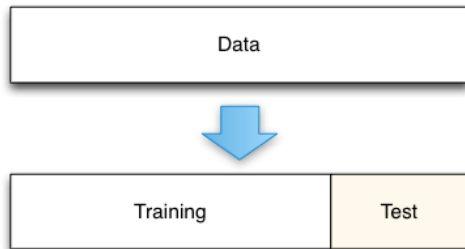
MNIST: 10 klasser for 0 – 9 siffer.
Målet er å identifisere hvilket siffer som er avbildet.



Klassifisering

Del dataen inn i såkalt **treningsdata** og **testdata**. Begge har tilhørende labels.

Under testing: kjør klassifiseringsalgoritmen på testdata som om labels var ukjent, og evaluer hvor godt modellen vår gjør det.



Score functions

Score functions: hvor godt scorer et datapunkt b for klasse k ?

$$\text{score}(b, k) = f_k(b), \quad f_k : \mathbb{R}^m \rightarrow \mathbb{R}. \quad (12)$$

Lav score \rightarrow Høy sannsynlighet for at et datapunkt b tilhører klasse k .

Klassifiser et datapunkt b til klasse $c(b)$ med lavest score:

$$c(b) = \arg \min_k f_k(b). \quad (13)$$

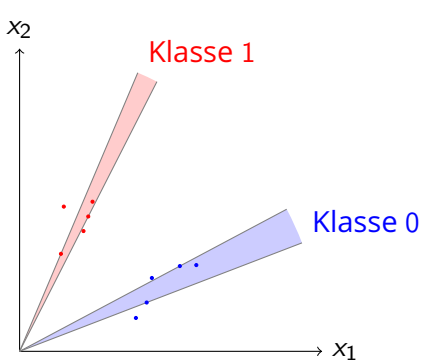
\rightarrow må beregne $f_k(b)$ for hver klasse.

Er det noen funksjoner vi har definert som kunne passe som score funksjon?

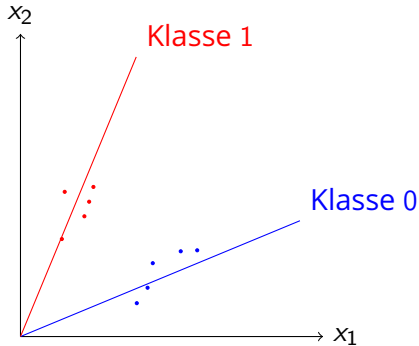
Er det noen funksjoner vi har definert som kunne passe som score funksjon?

Avstanden til en forhåndstrent dictionary $D_W(b)$!

Geometrisk interpretasjon av trening

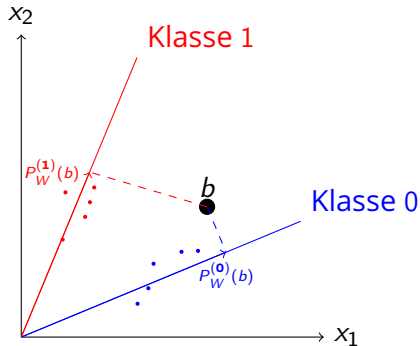


ENMF med $d = 2$, lær kjegler som best approssimerer data.



Truncated SVD med $d = 1$, lær underrom som best approssimerer data.

Geometrisk interpretasjon av testing



Observerer nytt punkt b : Projekter ned på de forskjellige dictionariene og klassifiser til nærmeste, som her er klasse 0.

Høyere dimensjoner og flere klasser: Mye mer komplisert.

Vår klassifiseringsalgoritme

1. **Trening:** Lær dictionaries $W^{(k)}$ på treningsdata med klasse k med forhåndsbestemt antall basisvektorer d .
2. **Testing:** For alle testdata b_i , beregn avstanden til alle dictionaries ved hjelp av projeksjon $D_W^{(k)}(b_i) = \|b_i - P_{W^{(k)}}(b_i)\|_2^2$.
Prediker klasse $c(b) = \arg \min_k D_{W^{(k)}}(b_i)$
3. **Evaluering:** Evaluer hvor godt modellen vår gjør det (se neste slide).
4. **Tilpassing:** Test og evaluer modellen for forskjellige parameter d .
5. Sammenlign og undersøk forskjellige modeller. Prøv å forstå hva modellen klarer å lære og hva de ikke klarer å lære.

Evaluering

For å evaluere klassifisering trenger vi kvantitative mål:

$$\text{accuracy} = \frac{\text{Antall riktige prediksjoner}}{\text{Totalt antall prediksjoner}}. \quad (14)$$

Godt mål for balanserte datasett. Ønsker at accuracy skal være så nærme 1 som mulig.

$$\text{recall} = \frac{\text{Antall riktige prediksjoner for denne klassen}}{\text{Antall data for denne klassen}}. \quad (15)$$

Klassifiseringsmetoder er ofte biased mot noen klasser, kan bruke recall for å undersøke dette.

Parameteret d

Antall basis vektorer i dictionaries d kan påvirke resultatene mye.

- ▶ Lav $d \rightarrow$ dictionaries er ikke fleksible nok til å rekonstruere data.
- ▶ Høy $d \rightarrow$ dictionaries for fleksible, og de kan rekonstruere data fra andre klasser.
- ▶ ENMF oppfører seg til annerledes siden vi er begrenset til ikke-negative basisvektorer...

Hyperparameter tilpassing: Velge god d (og andre parameter) slik at metoden(e) gir gode resultat \rightarrow Teste mange forskjellige d .

Samtidig: lav d er ønskelig siden beregningstid og lagringsplass øker med d . Her skal dere undersøke litt.

Oppgave 3

- ▶ Implementere vår foreslåtte klassifiseringsalgoritme.
- ▶ Teste dette på MNIST data.
- ▶ Undersøke hvor metodene feiler og hvor den gjør det godt.
- ▶ Undersøke hvordan metodene oppfører seg med parameteret d .
- ▶ Tilsammen rundt 30% av karakteren. Oppgaven går veldig raskt om dere har implementert oppgave 1 og 2 godt, ettersom dere kan og burde gjenbruke all kode.
- ▶ Sørg for at dere gjør oppgave 1 og 2 godt!

Mer om dette

Interessant prosjekt? Gå IndMat! Relevante fag:

- ▶ **TMA4268 Statistisk læring**
- ▶ **TMA4267 Lineære statistiske modeller**
- ▶ **TMA4180 Optimering I**
- ▶ TMA4145 Lineære metoder
- ▶ TMA4215 Numerisk matematikk
- ▶ TMA4205 Numerisk lineær algebra
- ▶ Datafag som TDT4120 Algoritmer og Datastrukturer.
- ▶ Unødvendig å ta datafag som Intro til AI og Maskinlæring...
- ▶ ... Skriv heller prosjekt og masteroppgave innen statistikk/optimering/numerik hvor dere bruker AI/maskinlæring.

Fullført utdanning → data scientist, veldig attraktivt yrke.