Introduksjon til maskinlæring

Bildegjenkjenning med Python og Tensorflow

Esten H. Leonardsen

23.09.24

UNIVERSITETET
I OSLO

Plan for dagen

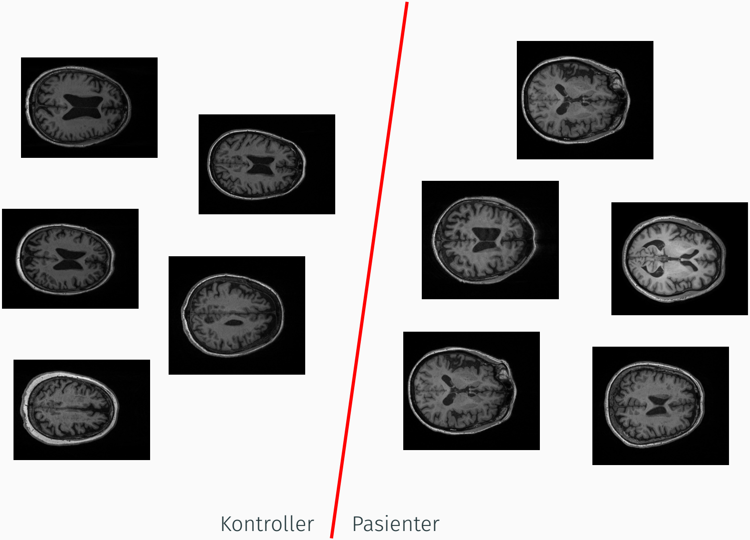
Teori:

- Hva er en statistisk læringsmodell?
- Hva er en kost-funksjon?
- Hvordan trener vi en statstisk læringsmodell?
- Hvordan fungerer et (dypt) kunstig nevralt nettverk?
- Hvordan fungerer et konvolusjonelt nevralt nettverk (for klassifikasjon)?
- Hva er transfer learning?
- Hva er overtilpasning, og hvordan unngår vi det?

Praktisk workshop:

1. Sette opp et Python-miljø i Google Colab
2. Predikere med et ferdigtrent konvolusjonelt nevralt nettverk
3. Retrene en klassifikator for blomsterarter
4. Hvis tid, forbedre klassifikatoren

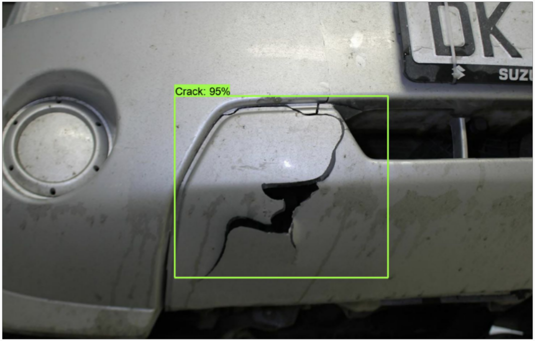
Motivasjon



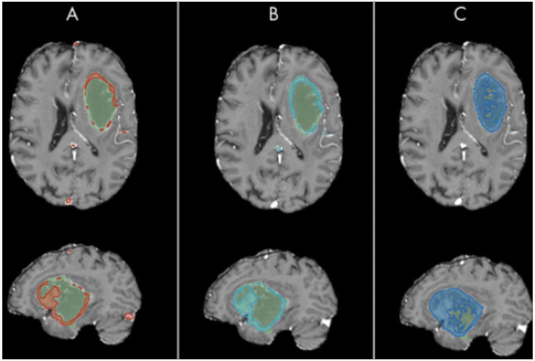
Motivasjon

https://drive.google.com/file/d/1CLgpFaXDukFpBmhHHzPCZdtuXtecXqPO/view

Motivasjon

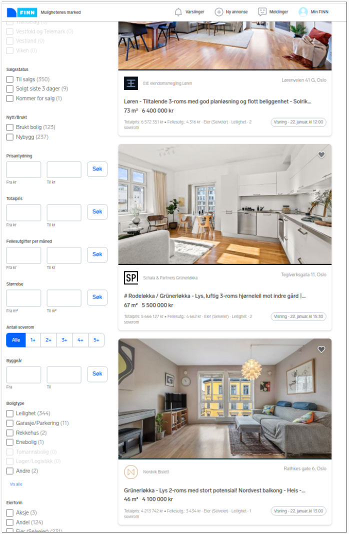


Motivasjon



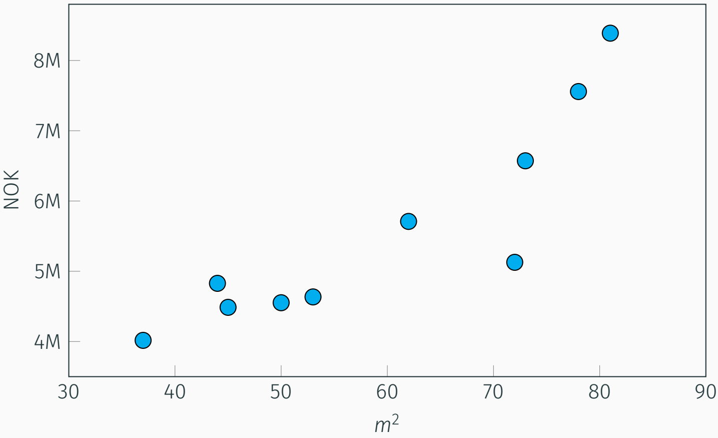
Motivasjon

Statistisk læring

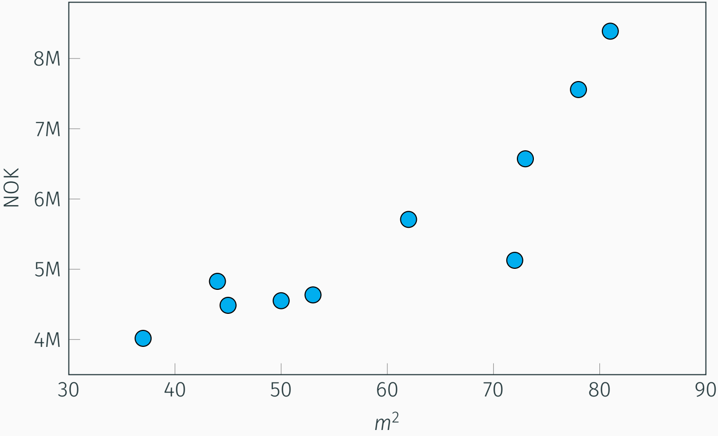

FINN
Mulighetenes marked
Varslinger
Ny annonse
Meldinger
Min FINN
Vestfold og Telemark (0)
Vestland (0)
Viken (0)
Salgsstatus
Til salgs (350)
Solgt siste 3 dager (9)
Kommer for salg (1)
Nytt/Brukt
Brukt bolig (123)
Nybygg (237)
Prisantydning
Søk
Fra kr
Til kr
Totalpris
Fellesutgifter per måned
Størrelse
Fra m²
Til m²
Antall soverom
Alle
1+
2+
3+
4+
5+
Byggeår
Fra
Til
Boligtype
Leilighet (344)
Garasje/Parkering (11)
Rekkehus (2)
Enebolig (1)
Tomannsbolig (0)
Lager/Logistikk (0)
Andre (2)
Vis alle
Eierform
Aksje (3)
Andel (124)
Lørenveien 41 G, Oslo
Løren - Tiltalende 3-roms med god planløsning og flott beliggenhet - Solrik...
73 m² 6 400 000 kr
Schala & Partners Grünerløkka
Teglverksgata 11, Oslo
Rodeløkka / Grünerløkka - Lys, luftig 3-roms hjørneleil mot indre gård |...
67 m² 5 500 000 kr
Nordvik Bislett
Rathkes gate 6, Oslo
Grünerløkka - Lys 2-roms med stort potensial! Nordvest balkong - Heis -...
46 m² 4 100 000 kr

m^2	**Price**
72	5.127.379
50	4.552.170
45	4.486.654
62	5.709.276
53	4.634.912
81	8.388.570
44	4.828.170
78	7.557.770
37	4.016.520
73	6.572.351

Statistisk læring: Modeller

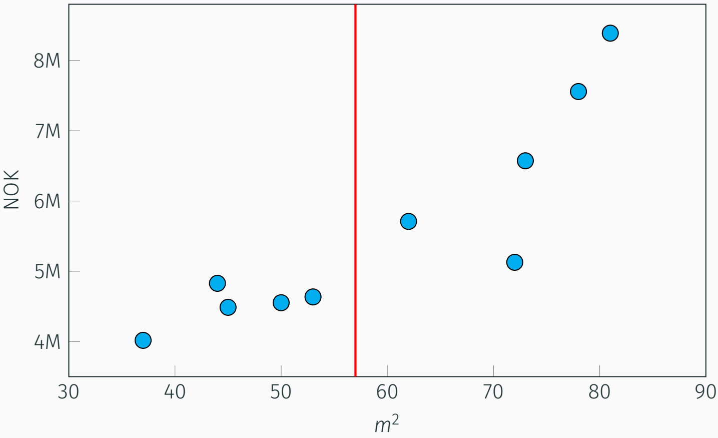


Statistisk læring: Modeller



$$\hat{y} = f(x)$$

Statistisk læring: Modeller

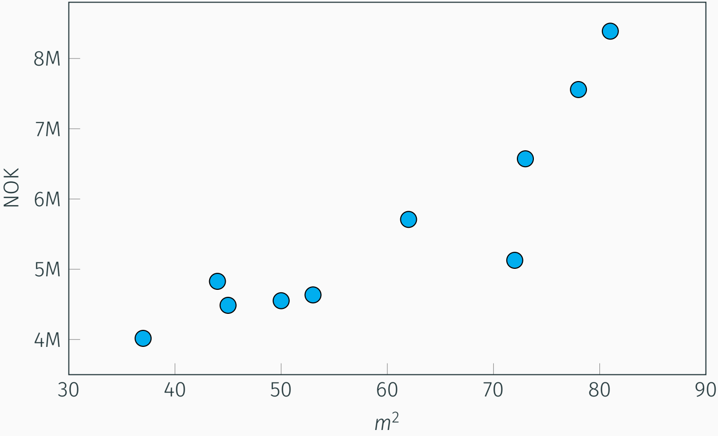


$$\hat{y} = f(x)$$

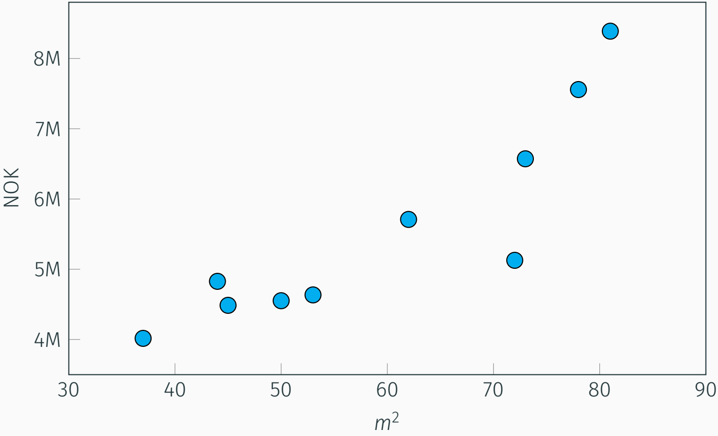
Hva er en statistisk læringsmodell?

- En matematisk konstruksjon (ofte en formel) som representerer forholdet mellom input x og output y
- En funksjonell enhet som, gitt ny data x er i stand til å predikere nye utfall y

Statistisk læring: Kost-funksjoner

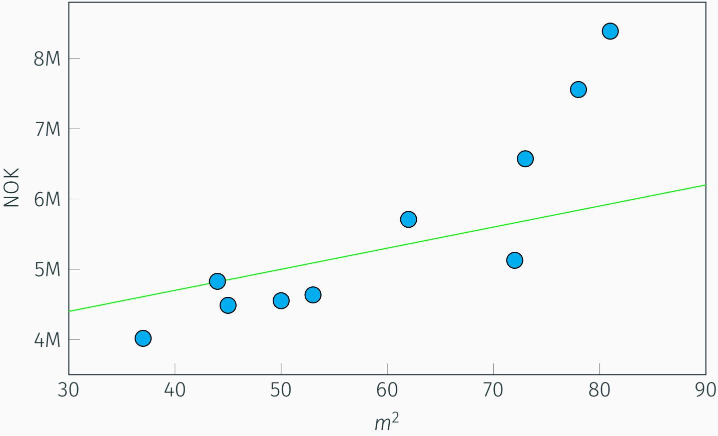


$$\hat{y} = b + wx$$



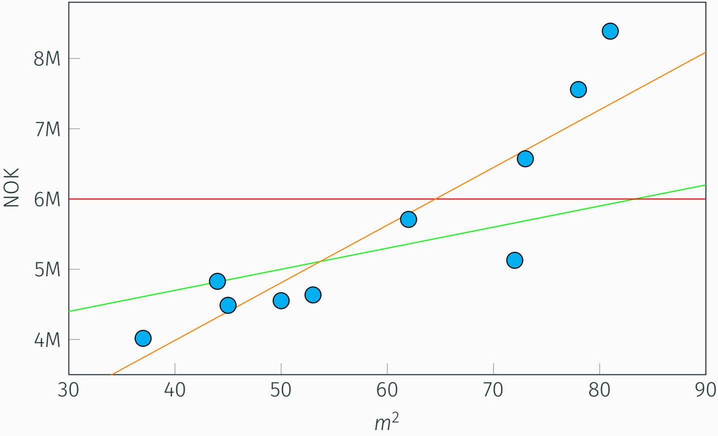
$$\hat{y} = b + wx$$

Statistisk læring: Kost-funksjoner

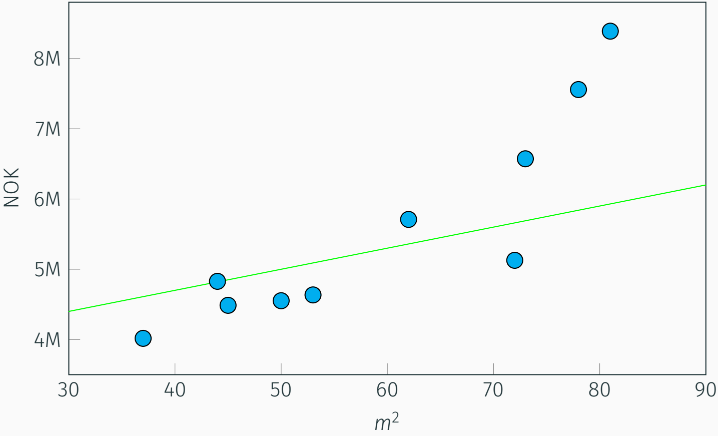


$\hat{y} = 3500000 + 30000x$

Statistisk læring: Kost-funksjoner

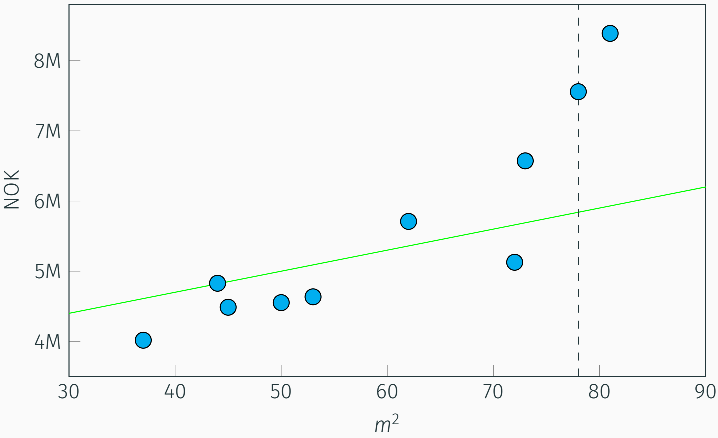


$\hat{y} = 6000000 + 0x$ $\hat{y} = 3500000 + 30000x$ $\hat{y} = 706495 + 82031x$

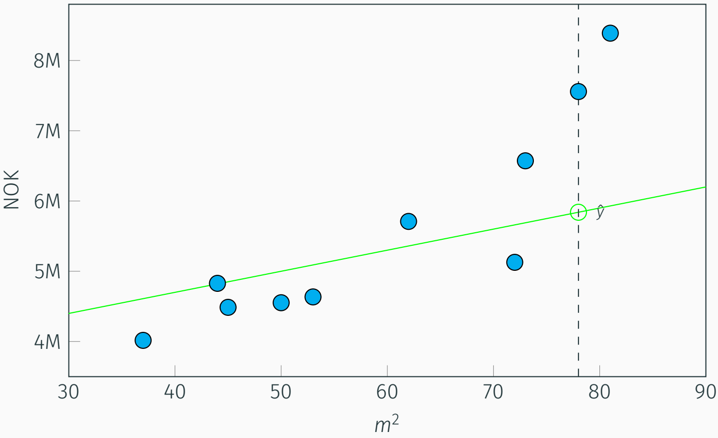


$$\hat{y} = 3500000 + 30000x$$

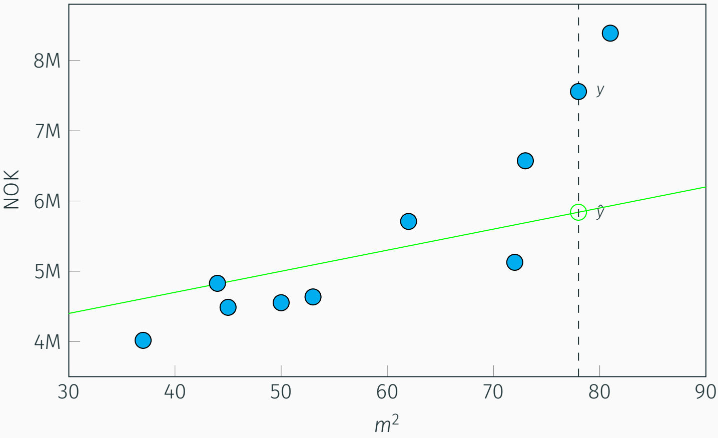
Statistisk læring: Kost-funksjoner



$$\hat{y} = 3500000 + 30000 * 78$$

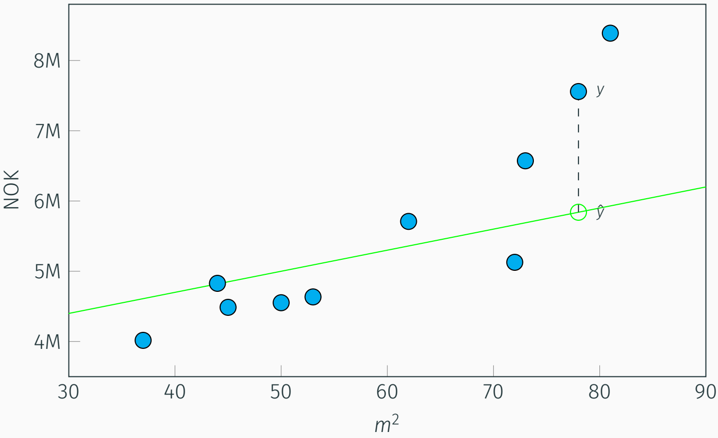


$$\hat{y} = 3500000 + 30000 * 78$$



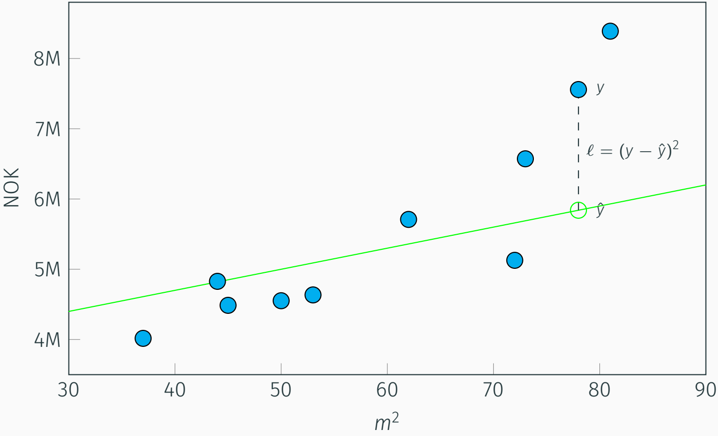
$$\hat{y} = 3500000 + 30000 * 78$$

Statistisk læring: Kost-funksjoner



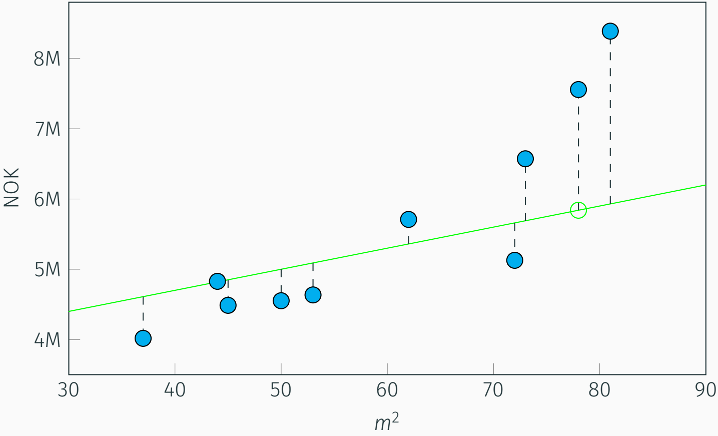
$\hat{y} = 3500000 + 30000 * 78$

Statistisk læring: Kost-funksjoner



$$\hat{y} = 3500000 + 30000 * 78$$

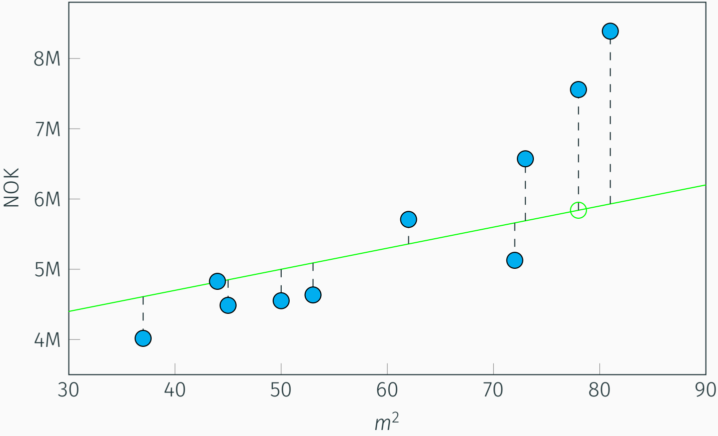
Statistisk læring: Kost-funksjoner



$$\hat{y} = 3500000 + 30000x$$

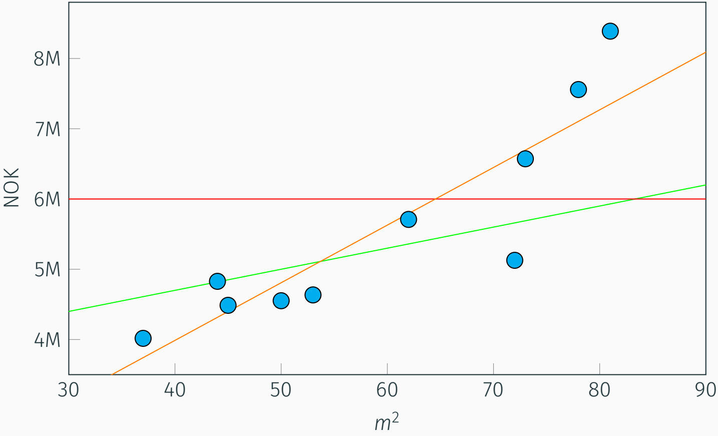
$$\ell = \frac{1}{N}\sum(y - \hat{y})^2$$

Statistisk læring: Kost-funksjoner



$$\hat{y} = 3500000 + 30000x$$
$$\ell = 1.10 \times 10^{13}$$

Statistisk læring: Kost-funksjoner



$\hat{y} = 6000000 + 0x$ $\ell = 2.08 \times 10^{13}$

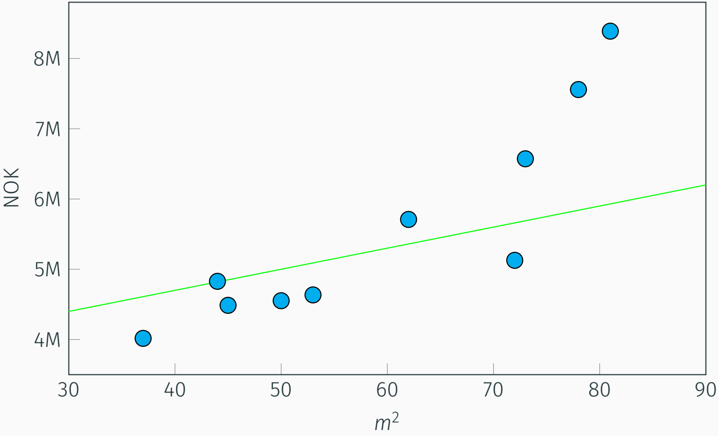
$\hat{y} = 3500000 + 30000x$ $\ell = 1.10 \times 10^{13}$

$\hat{y} = 706495 + 82031x$ $\ell = 4.09 \times 10^{12}$

Hva er en kost-funksjon?

- En matematisk funksjon som beregner kostnaden av å bruke en gitt modell for å forklare et gitt datasett
- En formalisering av oppgaven vi ønsker at modellen skal løse

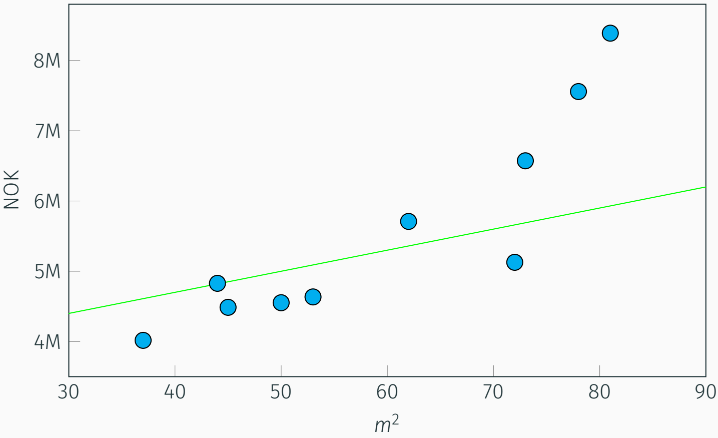
Statistisk læring: Trening



$$\hat{y} = b + wx$$

$$\ell = \frac{1}{N} \sum (y - \hat{y})^2$$

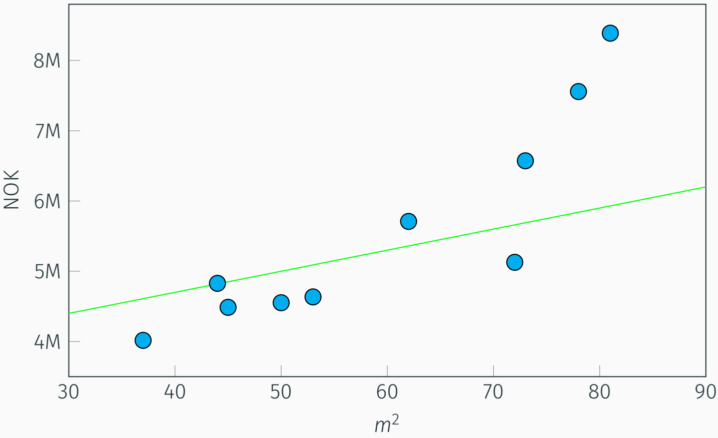
Statistisk læring: Trening



$$\boxed{\hat{y}} = b + wx$$

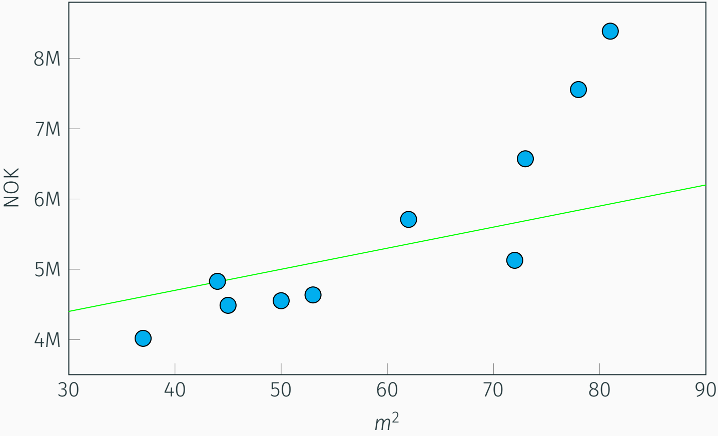
$$\ell = \frac{1}{N}\sum(y - \boxed{\hat{y}})^2$$

Statistisk læring: Trening



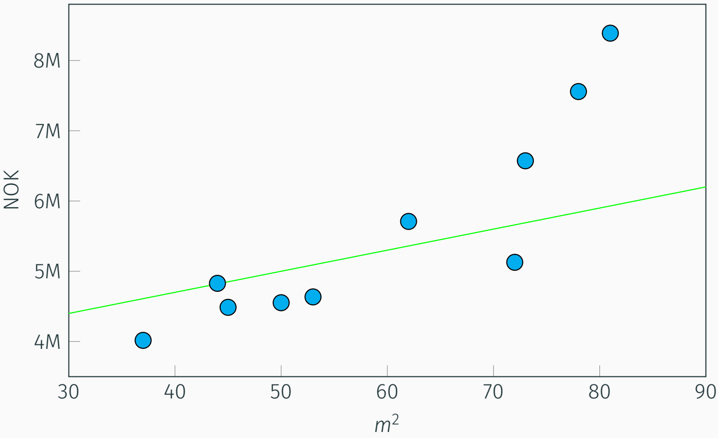
$$\ell = \frac{1}{N}\sum(y-(b+wx))^2$$

Statistisk læring: Trening



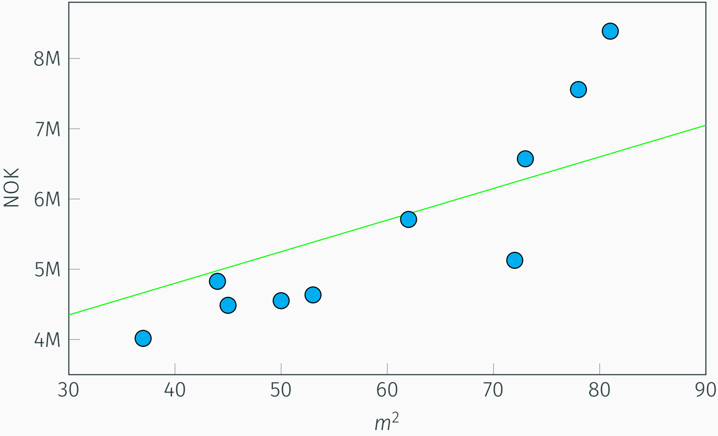
$$1.10 \times 10^{13} = \frac{1}{N}\sum(y - (3500000 + 30000x))^2$$

Statistisk læring: Trening



$$1.10 \times 10^{13} = \frac{1}{N}\sum(y - (3500000 + 30000x))^2$$

Statistisk læring: Trening



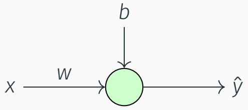
$$7.24 \times 10^{12} = \frac{1}{N}\sum(y - (3000000 + 45000x))^2$$

Statistisk læring: Trening

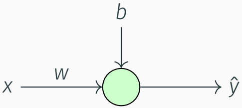
Hvordan trener vi en statistisk læringsmodell?

1. Estimer prediksjoner for alle datapunkter
2. Beregn kostnaden av disse prediksjonene (ved å sammenligne med fasit)
3. Kalkuler hvordan parametrene i modellen påvirker kostnaden
4. Oppdater parametrene i den retningen som minsker kostnaden

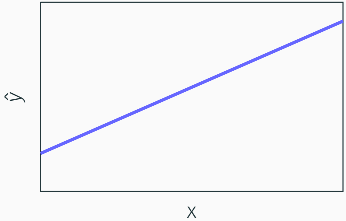
Kunstige nevrale nettverk



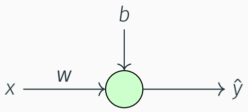
$$\hat{y} = wx + b$$



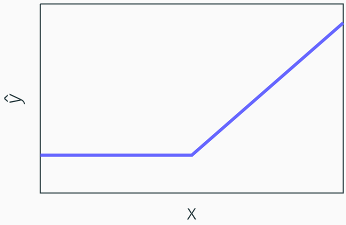
$$\hat{y} = wx + b$$

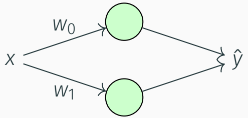


Kunstige nevrale nettverk: Oppbygning

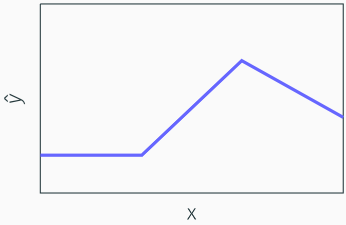


$$\hat{y} = max(0, wx + b)$$





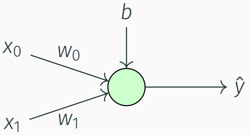
$$\hat{y} = max(0, w_0x + b_0) + max(0, w_1x + b_1)$$



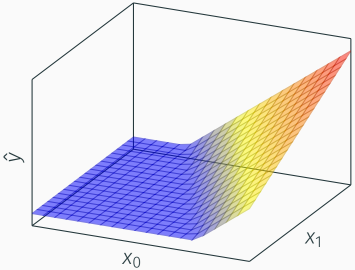
Universal approximation theorem:

"Any relationship that can be described with a polynomial function can be approximated by a neural network with a single hidden layer."

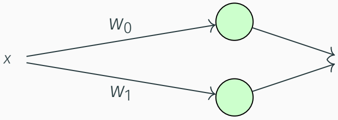
Kunstige nevrale nettverk: Oppbygning



$$\hat{y} = max(0, w_0 x_0 + w_1 x_1 + b)$$

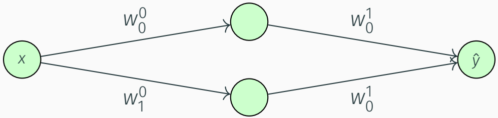


Kunstige nevrale nettverk: Oppbygning

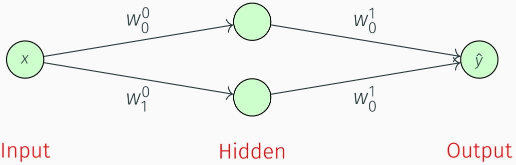


$$\hat{y} = max(0, w_0x + b_0) + max(0, w_1x + b_1)$$

Kunstige nevrale nettverk: Oppbygning

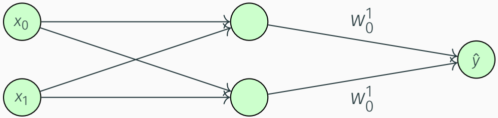


$$\hat{y} = max(0, w_{0,0}^1 * max(0, w_{0,0}^0 * x + b_{0,0}) + w_{1,0}^1 * max(0, w_{0,1}^0 * x + b_{1,0}) + b_1)$$



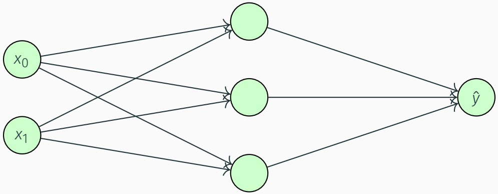
$$\hat{y} = max(0, w^1_{0,0} * max(0, w^0_{0,0} * x + b_{0,0}) + w^1_{1,0} * max(0, w^0_{0,1} * x + b_{1,0}) + b_1)$$

Kunstige nevrale nettverk: Oppbygning



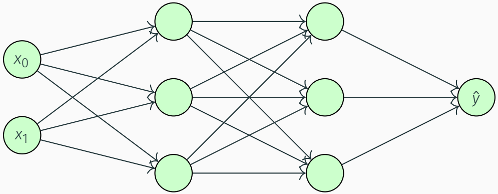
$$\hat{y} = max(0, w^1_{0,0} * max(0, w^0_{0,0} * x_0 + w^0_{1,0} * x_1 + b_{0,0}) + w^1_{1,0} * max(0, w^0_{0,1} * x_0 + w^0_{1,1} * x_1 + b_{0,1}) + b_1)$$

Kunstige nevrale nettverk: Oppbygning



$$
\begin{aligned}
\hat{y} = max(0, & w^1_{0,0} * max(0,\ w^0_{0,0} * x_0 + w^0_{1,0} * x_1 + b_{0,0}) + \\
& w^1_{1,0} * max(0,\ w^0_{0,1} * x_0 + w^0_{1,1} * x_1 + b_{0,1}) + \\
& w^1_{2,0} * max(0,\ w^0_{0,2} * x_0 + w^0_{1,2} * x_1 + b_{0,2}) + \\
& b_1)
\end{aligned}
$$

Kunstige nevrale nettverk: Oppbygning



$$
\begin{aligned}
\hat{y} = max(0, & w^2_{0,0} * max(0, w^1_{0,0} * max(0, w^0_{0,0} * x_0 + w^0_{1,0} * x_1 + b_{0,0}) + \\
& \qquad w^1_{1,0} * max(0, w^0_{0,1} * x_0 + w^0_{1,1} * x_1 + b_{0,1}) + \\
& \qquad w^1_{2,0} * max(0, w^0_{0,2} * x_0 + w^0_{1,2} * x_1 + b_{0,2}) + \\
& \qquad b_{1,0}) + \\
& w^2_{1,0} * max(0, w^1_{0,1} * max(0, w^0_{0,0} * x_0 + w^0_{1,0} * x_1 + b_{0,0}) + \\
& \qquad w^1_{1,1} * max(0, w^0_{0,1} * x_0 + w^0_{1,1} * x_1 + b_{0,1}) + \\
& \qquad w^1_{2,1} * max(0, w^0_{0,2} * x_0 + w^0_{1,2} * x_1 + b_{0,2}) + \\
& \qquad b_{1,1}) + \\
& w^2_{2,0} * max(0, w^1_{0,2} * max(0, w^0_{0,0} * x_0 + w^0_{1,0} * x_1 + b_{0,0}) + \\
& \qquad w^1_{1,2} * max(0, w^0_{0,1} * x_0 + w^0_{1,1} * x_1 + b_{0,1}) + \\
& \qquad w^1_{2,2} * max(0, w^0_{0,2} * x_0 + w^0_{1,2} * x_1 + b_{0,2}) + \\
& \qquad b_{1,2}) + \\
& b_2)
\end{aligned}
$$

Kunstige nevrale nettverk: Oppbygning

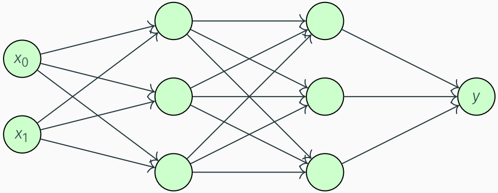
Hvordan fungerer et (dypt) kunstig nevralt nettverk?

- Kunstige nevroner som hver for seg beregner en enkel, ikke-lineær funksjon kombineres i en komputasjonell graf
- Grafen som helhet kan beregne arbitrært komplekse sammenhenger mellom input og output

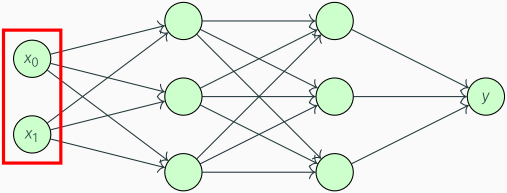
Konvolusjonelle nevrale nettverk

UNIVERSITETET
I OSLO

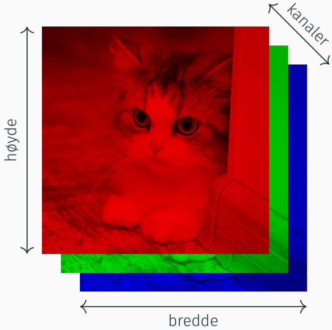
Konvolusjonelle nevrale nettverk: Bildeinput



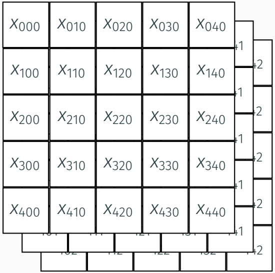
Konvolusjonelle nevrale nettverk: Bildeinput



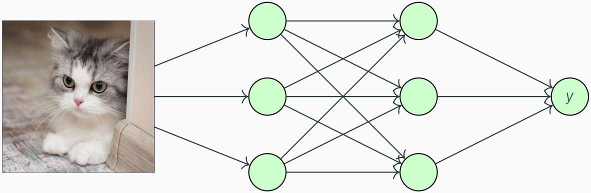
Konvolusjonelle nevrale nettverk: Bildeinput


kanaler
høyde
bredde

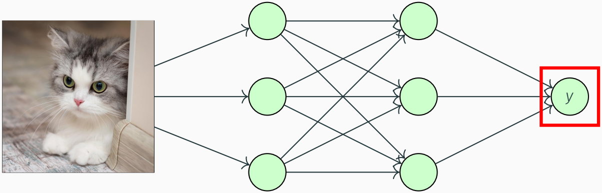
Konvolusjonelle nevrale nettverk: Bildeinput



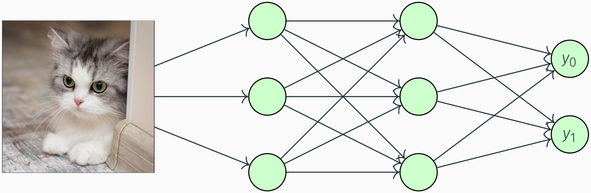
Konvolusjonelle nevrale nettverk: Bildeinput



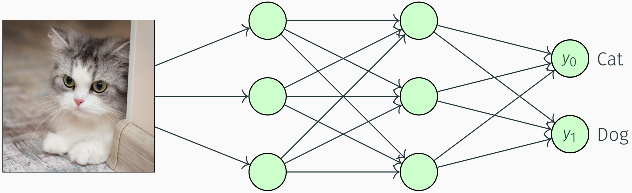
Konvolusjonelle nevrale nettverk: Klassifikasjonsoutput

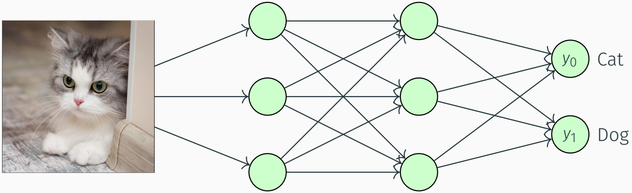


Konvolusjonelle nevrale nettverk: Klassifikasjonsoutput



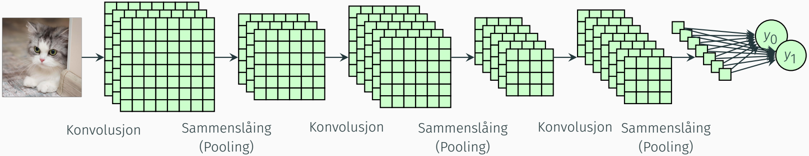
Konvolusjonelle nevrale nettverk: Klassifikasjonsoutput





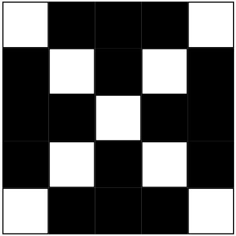
$$\text{kost} = -\sum_i y_i \log(\hat{y}_i)$$

Konvolusjonelle nevrale nettverk: Arkitektur



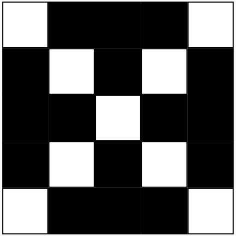
Konvolusjonelle nevrale nettverk: Konvolusjon

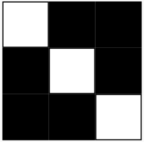
Bilde



Konvolusjonelle nevrale nettverk: Konvolusjon

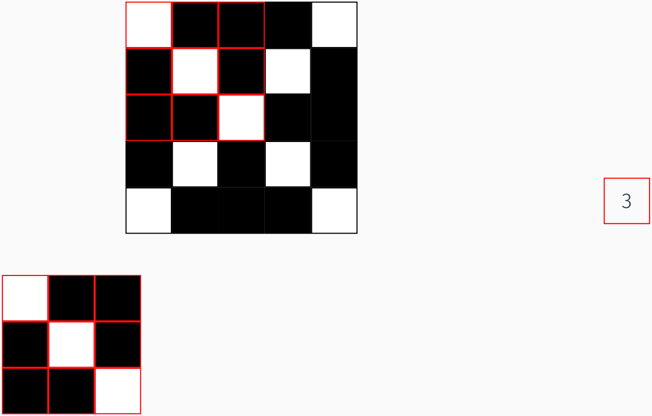
Bilde



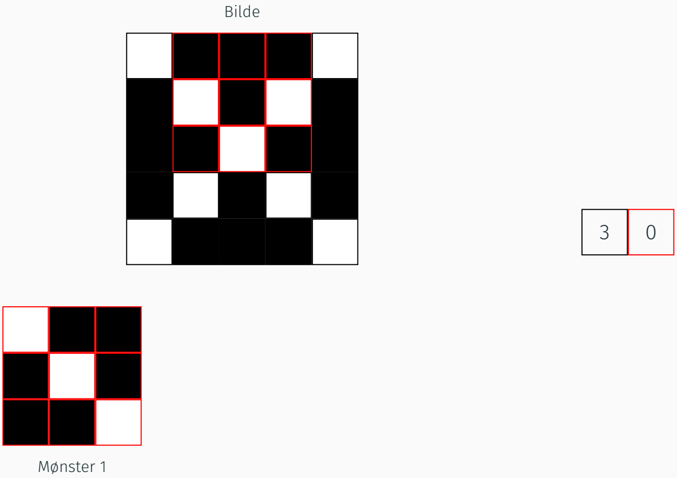


Mønster 1

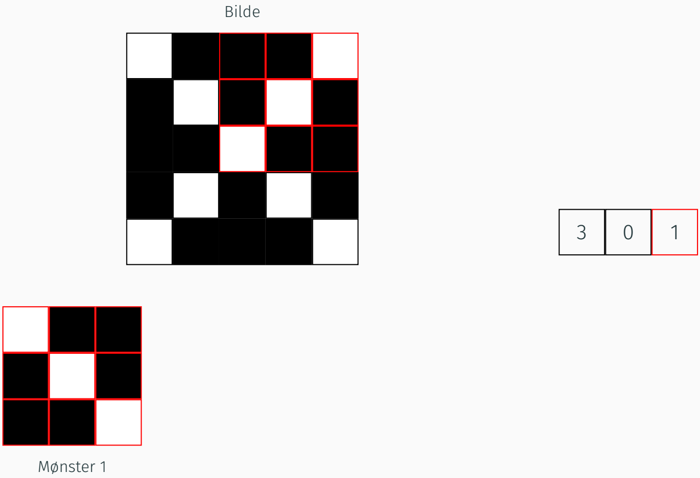
Bilde



Mønster 1

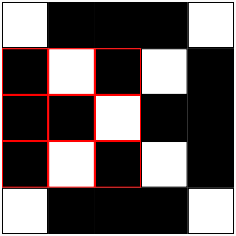

Bilde
3
0
Mønster 1

Konvolusjonelle nevrale nettverk: Konvolusjon

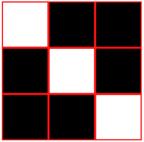


Konvolusjonelle nevrale nettverk: Konvolusjon

Bilde



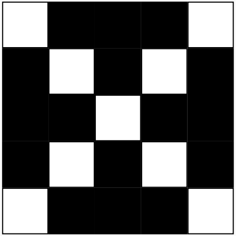
3	0	1
0		



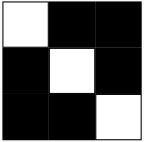
Mønster 1

Konvolusjonelle nevrale nettverk: Konvolusjon

Bilde



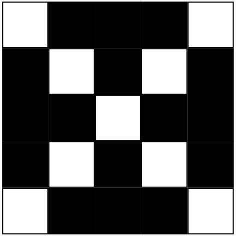
3	0	1
0	3	0
1	0	3

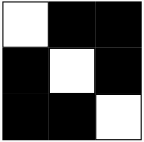


Mønster 1

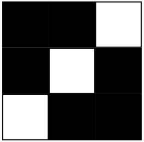
Konvolusjonelle nevrale nettverk: Konvolusjon

Bilde





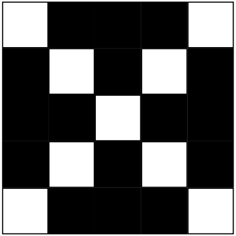
Mønster 1

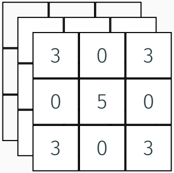


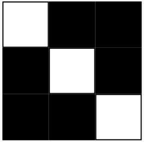
Mønster 2

1	0	3
0	3	0
3	0	1

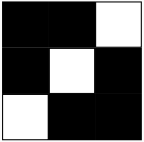
Bilde



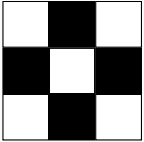




Mønster 1

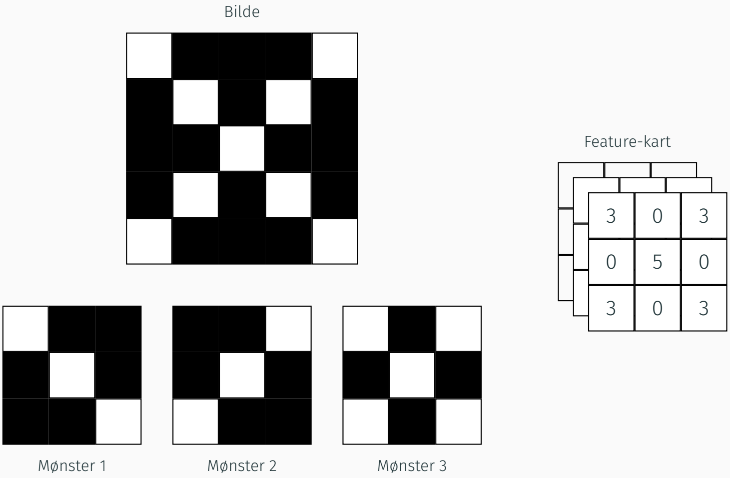


Mønster 2

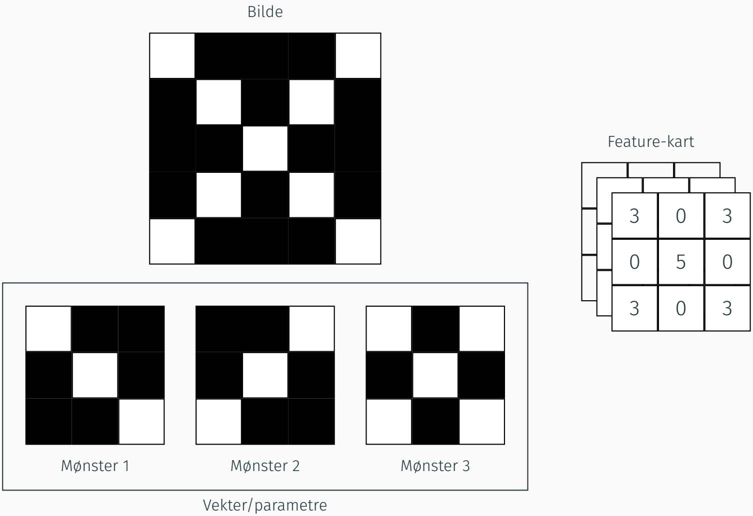


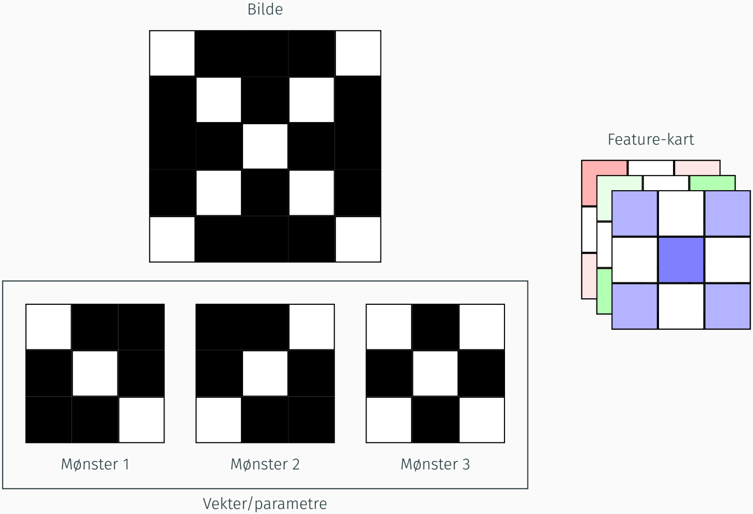
Mønster 3

Konvolusjonelle nevrale nettverk: Konvolusjon



Konvolusjonelle nevrale nettverk: Konvolusjon




Bilde
Feature-kart
Mønster 1
Mønster 2
Mønster 3
Vekter/parametre

Konvolusjonelle nevrale nettverk: Pooling

Feature-kart

0	1	2	3
4	5	6	7
8	9	10	1
12	13	14	15

Feature-kart

0	1	2	3
4	5	6	7
8	9	10	1
12	13	14	15

5

Konvolusjonelle nevrale nettverk: Pooling

Feature-kart

0	1	2	3
4	5	6	7
8	9	10	1
12	13	14	15

5	7

Konvolusjonelle nevrale nettverk: Pooling

Feature-kart

0	1	2	3
4	5	6	7
8	9	10	1
12	13	14	15

5	7
13	

Feature-kart

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15

5	7
13	15

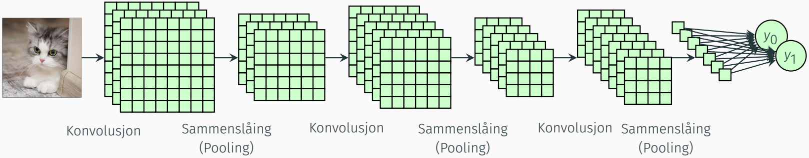
Konvolusjonelle nevrale nettverk: Pooling

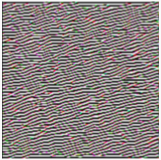
Feature-kart

0	1	2	3
4	5	6	7
8	9	10	1
12	13	14	15

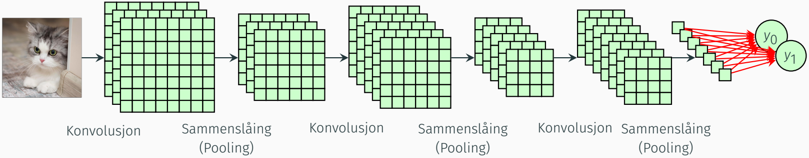
5	7
13	15

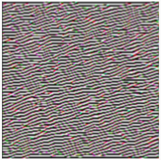
Konvolusjonelle nevrale nettverk: Arkitektur



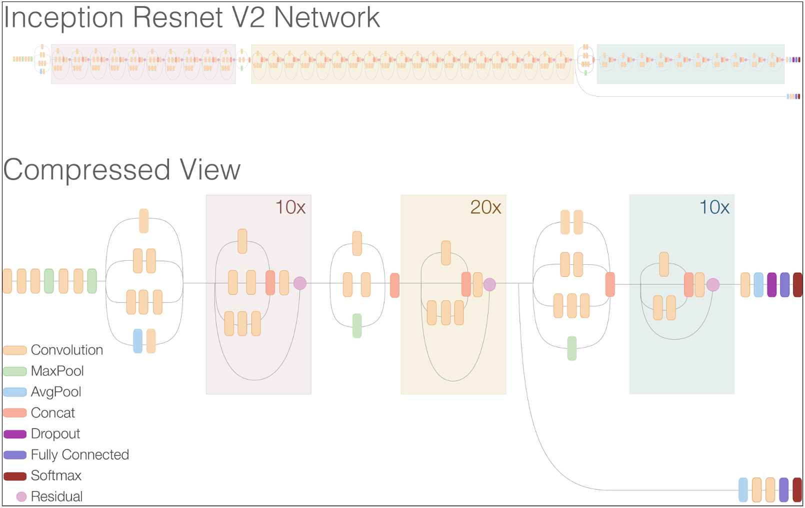


Konvolusjonelle nevrale nettverk: Arkitektur





Konvolusjonelle nevrale nettverk: Arkitektur



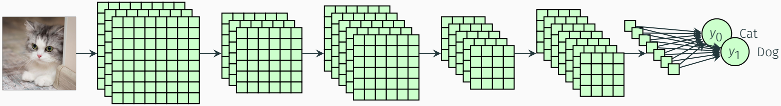
Konvolusjonelle nevrale nettverk: Arkitektur

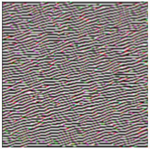
Hvordan fungerer et konvolusjonelt nevralt nettverk (for klassifikasjon)?

- Bilder representeres ved 3-dimensjonale arrayer i en datamaskin, som kan mates direkte som input til nettverket
- For å oppnå klassifikasjon gir vi nettverket flere output-nevroner, en for hver klasse vi er interessert i, også tolker vi outputen slik at nevronet med høyest verdi sammensvarer med den predikerte klassifikatoren
- Inne i nettverket utføres alternerende konvolusjoner og pooling som i praksis lar oss gjenkjenne større og mer abstrakte mønstre jo dypere i modellen vi kommer
- Til slutt kobles mønstrene opp mot klassene ved et ”vanlig” nevralt nett (alle-til-alle koblinger)
- Parametrene som trenes ligger i konvolusjonene og sier hvilke mønstre som må læres, og i det siste laget

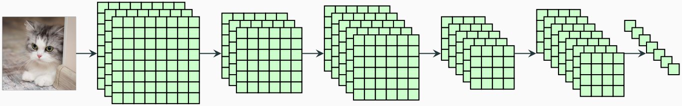
Praktiske tips

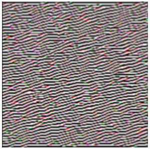
Praktiske tips: Transfer learning



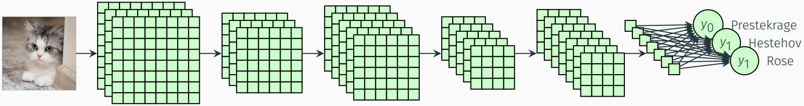


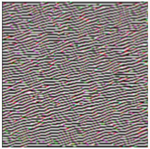
Praktiske tips: Transfer learning



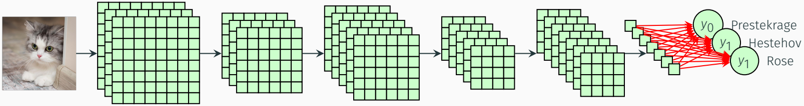


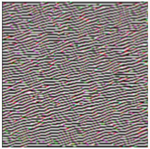
Praktiske tips: Transfer learning



Praktiske tips: Transfer learning



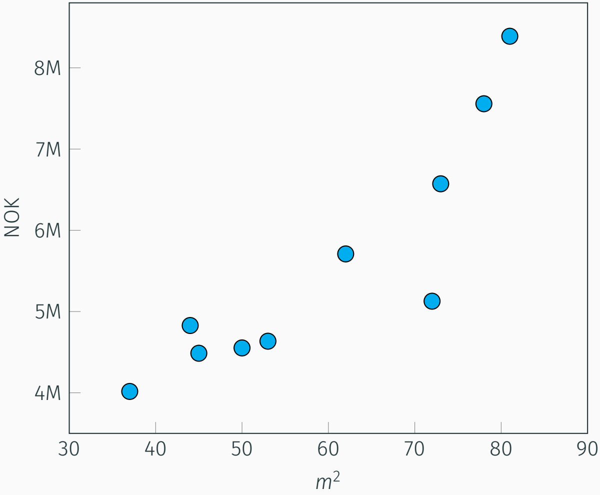


Praktiske tips: Transfer learning

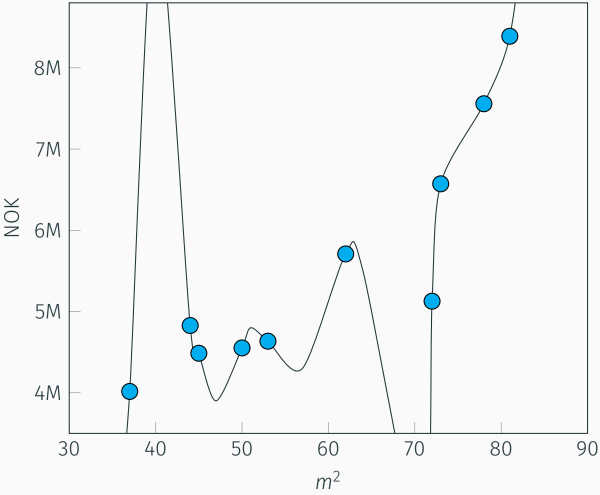
Hva er transfer learning?

- En teknikk for å utnytte at det er overlapp mellom visuelle problemer, og at et nettverk som har blitt trent til å løse èn oppgave antageligvis har lært noe som er nyttig også for en annen
- I praksis: Vi beholder de første lagene av en ferdigtrent modell (som gjenkjenner generiske visuelle mønstre, kanter, farger, geometri etc) og trener nye lag på toppen av disse

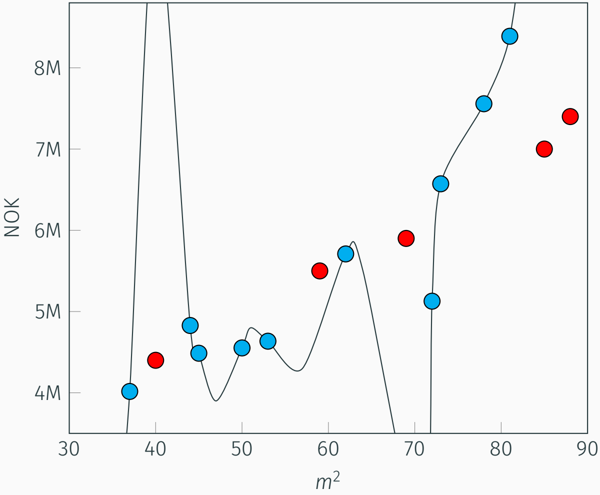
Praktiske tips: Overtilpasning



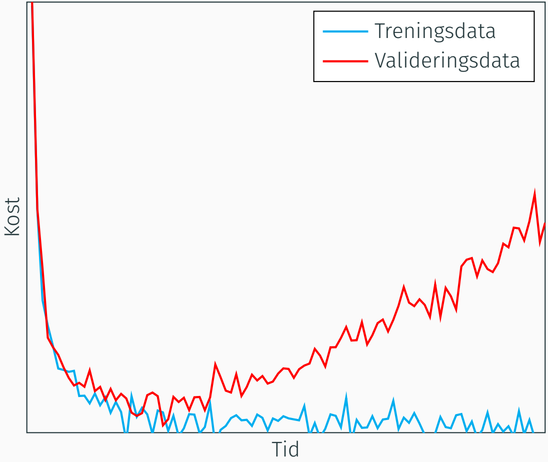
Praktiske tips: Overtilpasning



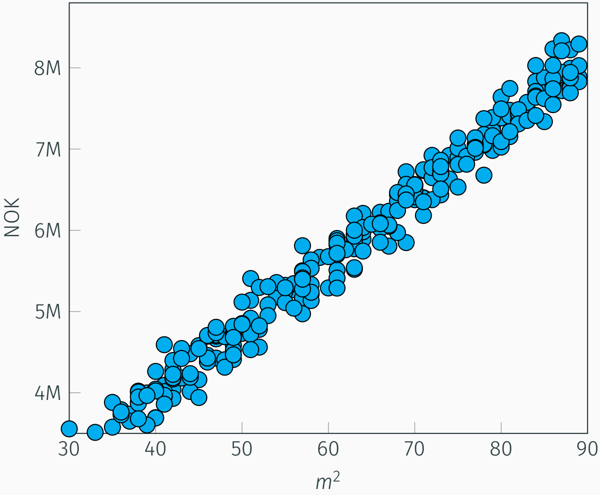
Praktiske tips: Overtilpasning



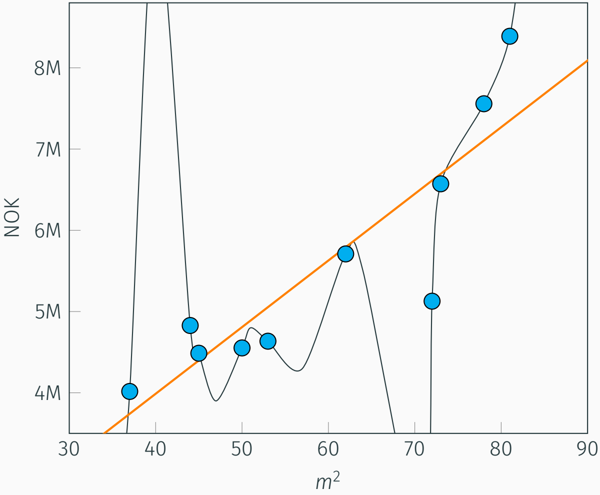
Praktiske tips: Overtilpasning



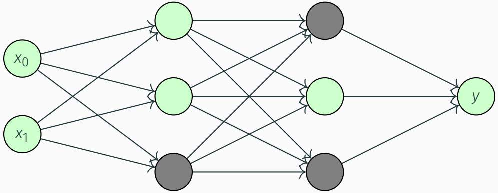
Praktiske tips: Overtilpasning



Praktiske tips: Overtilpasning

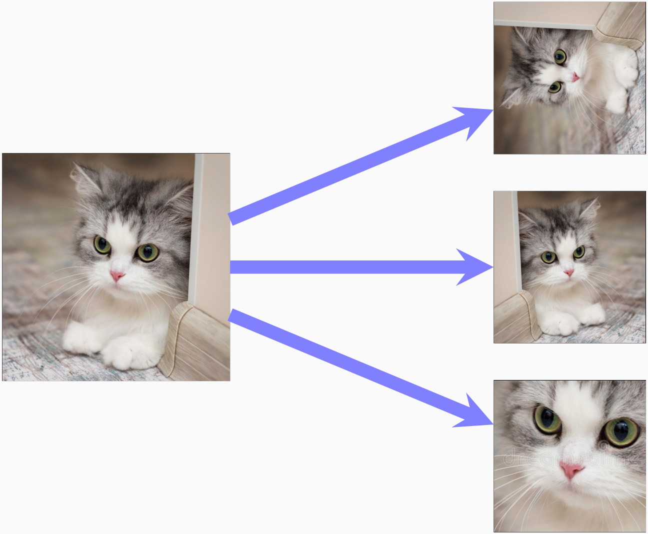


Praktiske tips: Overtilpasning

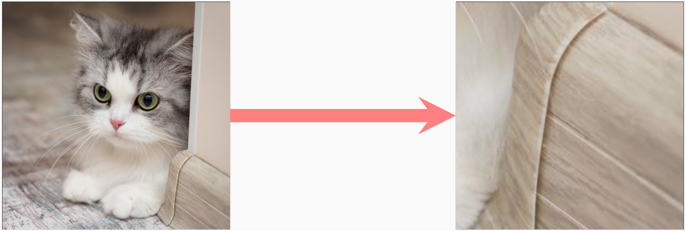


Praktiske tips: Overtilpasning

Praktiske tips: Overtilpasning



Praktiske tips: Overtilpasning



Praktiske tips: Overtilpasning

Hva er overtilpasning, og hvordan unngår vi det?

- Modellen har lært å gjenkjenne mønstre i treningsdataen som **ikke holder** i det generelle tilfellet (e.g. i ny data)
- Vi unngår det ved: Store datamengder, strikt testing (**!!!**), regularisering, augmentering, ...