MAT1080 H25
Oscar Brovold

Løsningsforslag kapittel 2

Oppgaver relatert til maskinlæring

Oppgave 33a

Vi skal finne den deriverte til sigmoidfunksjonen:

$$\sigma(x) = \frac{1}{1+e^{-x}}.$$

Vi har:

$$\sigma(x) = \frac{1}{1+e^{-x}} = (1+e^{-x})^{-1}.$$

Finner den deriverte ved kjerneregelen:

$$\begin{aligned}
\sigma'(x) &= -(1+e^{-x})^{-2} \cdot -e^{-x} \\
&= \frac{1}{e^x}\frac{1}{(1+e^{-x})^2} \\
&= \underbrace{\tfrac{1}{e^x}\tfrac{1}{(1+e^{-x})}}_{1-\sigma(x)} \quad \underbrace{\tfrac{1}{(1+e^{-x})}}_{\sigma(x)}
\end{aligned}$$

Vi har derfor $\sigma'(x) = \sigma(x)(1-\sigma(x))$.

Oppgave 33b

Vi skal finne den deriverte av den hyperbolske tangensfunksjonen:

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}.$$

Vi har:

$$\begin{aligned}
\tanh'(x) &= \frac{(e^x+e^{-x})(e^x+e^{-x}) - (e^x-e^{-x})(e^x-e^{-x})}{(e^x+e^{-x})^2} \\
&= \frac{(e^x+e^{-x})^2 - (e^x-e^{-x})^2}{(e^x+e^{-x})^2} \\
&= 1 - \tanh^2(x).
\end{aligned}$$

Totalt har vi derfor $\tanh'(x) = 1 - \tanh^2(x)$.

Oppgave 33c

Vi skal finne den deriverte av softplus-funksjonen:

$$\text{softplus}(x) = s(x) = \ln(1 + e^x).$$

Vi har da:

$$s'(x) = \frac{1}{1 + e^x} e^x = \sigma(x).$$

$\sigma(x)$ er her sigmoidfunksjonen vi jobbet med i a).

Oppgave 33d

Vi skal finne den deriverte til tapsfunksjonen kjent som "Binary Cross Entropy (BCE)":

$$L_{BCE}(x) = -(y \ln x + (1 - y) \ln(1 - x)).$$

Vi har:

$$L_{BCE}(x) = -y \ln x - (1 - y) \ln(1 - x).$$

$$\begin{aligned} L'_{BCE}(x) &= -y\frac{1}{x} - (1 - y)\frac{1}{1 - x} \cdot -1 \\ &= -\frac{y}{x} + \frac{1 - y}{1 - x} \\ &= -\frac{y(1 - x)}{x(1 - x)} + \frac{x(1 - y)}{x(1 - x)} \\ &= \frac{-y + xy + x - xy}{x(1 - x)} \\ &= \frac{x - y}{x(1 - x)}. \end{aligned}$$

Totalt har vi da $L'_{BCE}(x) = \frac{x - y}{x(1 - x)}$.

Oppgave 34a

Vi skal i denne oppgaven starte med se på den kjente aktiveringsfunksjonen fra maskinlæring ReLU (Rectified linear unit):

$$\text{ReLU}(x) = \begin{cases} x, & x > 0, \\ 0, & x \leq 0. \end{cases}$$

For $x > 0$ vil ReLU fungere som identitetsfunksjonen. For $x \leq 0$ returnerer funksjonen 0. Altså lar den bare positive verdier passere og setter negative til null.

Den deriverte blir:

$$\text{ReLU}'(x) = \begin{cases} 1, & x > 0, \\ 0, & x < 0. \end{cases}$$

Formelt er ReLU ikke deriverbar i $x = 0$, siden den deriverte fra høyre og venstre side ikke er like. I praksis, når vi implementerer dette i datavitenskap/ML, løser vi dette ved å definere den deriverte i $x = 0$ som enten 1 eller 0. Konvensjonen er ofte at den deriverte i $x = 0$ er 0.

Oppgave 34b

Vi ser i denne oppgaven på en glatt tilnærming til ReLU-funksjonen, nemlig softplus-funksjonen, gitt ved:

$$s(x) = \ln(1 + e^x).$$

Vi skal i denne oppgaven finne den deriverte, og vise at den deriverte er lik sigmoidfunksjonen, altså:

$$s'(x) = \sigma(x) = \frac{1}{1 + e^{-x}}.$$

Vi bruker her kjerneregelen på $s(x)$, og resultatet kommer mer eller mindre direkte:

$$\begin{aligned} s'(x) &= \frac{1}{1 + e^x} e^x \\ &= \frac{1}{1 + e^{-x}} = \sigma(x). \end{aligned}$$

Oppgave 34c

For å finne den andre deriverte til $s(x)$ deriverer vi $\sigma(x)$. Vi får da:

$$\begin{aligned} \sigma'(x) &= -(1 + e^{-x})^{-2} \cdot -e^{-x} \\ &= \frac{1}{e^x} \frac{1}{(1 + e^{-x})^2} \\ &= \underbrace{\tfrac{1}{e^x} \tfrac{1}{(1+e^{-x})}}_{1-\sigma(x)} \quad \underbrace{\tfrac{1}{(1+e^{-x})}}_{\sigma(x)} \end{aligned}$$

Vi har derfor $s''(x) = \sigma'(x) = \sigma(x)(1 - \sigma(x))$.

Oppgave 34d

For å finne ut hvor $s(x)$ vokser og hvor den avtar, kan vi se på den deriverte. Vi har:

$$s'(x) = \sigma(x).$$

Siden $0 < \sigma(x) < 1$ for alle x, har vi $s'(x) > 0$. Altså er $s(x)$ er en strengt voksende funksjon.

Vi kan studere krumning til $s(x)$ ved å se på den andrederiverte, vi har:

$$s''(x) = \sigma(x)(1 - \sigma(x)).$$

Altså har vi $s''(x) > 0$, og $s(x)$ krummer derfor oppover (Ev. $s(x)$ er en konveks funksjon).

Oppgave 35

Denne oppgaven tar for seg et eksempel på hvordan vi kan trene et nevralt nettverk. Eksempelet er minimalt, hvor vi har en lineær funksjon etterfulgt av en aktiveringsfunksjon, og til slutt en tapsfunksjon. Spesifikt ser vi på denne lineære funksjonen:

$$z = wx + b.$$

z er her en funksjon som avhenger av x, hvor w og b er konstanter som vi ønsker at maskinen skal lære. Videre har vi en aktiveringsfunksjon, det kan f.eks. være sigmoid, ReLU eller softplus. Denne tar inn utgangsverdien til den lineære funksjonen, og produserer en ikke-lineær variant. Altså har vi:

$$a = s(z).$$

Til slutt har vi:

$$L = (a - y)^2.$$

Dette er en såkalt kvadratisk tapsfunksjon, og måler hvor stor "feil" modellen vår gjorde.

Målet med oppgaven er å finne de partielle deriverte, som til sammen utgjør gradienten til tapsfunksjonen. Gradienten forteller oss i hvilken retning funksjonen L vokser raskest, og er i dette tilfellet gitt ved:

$$\nabla L(w,b) = \left(\frac{\partial L}{\partial w}, \frac{\partial L}{\partial b}\right).$$

Ved å ta et steg i parameterrommet (w,b) i motsatt retning av gradienten vil vi få størst nedgang i L. Vi må derfor finne de partielle deriverte, som er det oppgaven spør oss om å finne.

I denne oppgaven bruker jeg såkalt Leibniz notasjon for kjernereglen, som er mye brukt i maskinlæring. Det er nøyaktig den samme som er introdusert i kompendiet, bare skrevet på en litt annen måte. Fordelen er at den gjør det lettere å holde styr på lange sekvenser av sammensatte funksjoner.

Vi får:

$$\frac{\partial L}{\partial w} = \frac{\partial L}{\partial a}\frac{\partial a}{\partial z}\frac{\partial z}{\partial w} \qquad \text{og} \qquad \frac{\partial L}{\partial b} = \frac{\partial L}{\partial a}\frac{\partial a}{\partial z}\frac{\partial z}{\partial b}.$$

Hvor

$$\frac{\partial L}{\partial a} = 2(a-y), \qquad \frac{\partial a}{\partial z} = s'(z), \qquad \frac{\partial z}{\partial w} = x \qquad \text{og} \qquad \frac{\partial z}{\partial b} = 1.$$

Vi får da resultatene oppgaven spør om:

$$\frac{\partial L}{\partial w} = 2\,(a-y)\,s'(z)\,x,$$

og

$$\frac{\partial L}{\partial b} = 2\,(a-y)\,s'(z).$$

Bemerkning: Vi kunne derivert $L(w,b) = (s(wx+b)-y)^2$ direkte, og endt opp med nøyaktig samme resultat. Leibniz notasjonen bare faktoriserer i meningsfulle ledd som skalerer seg bra i dype nevrale nettverk.