Oppgaver med løsningsforslag - SOK3023

Maskinlæring for økonomer
Laget av Markus J. Aase

January 15, 2025

Dette dokumentet inneholder oppgaver studentene i SOK-3023 kan jobbe med etter første uka av kurset. En del av svarene vil dere kunne finne fra de fysiske forelesningene og/eller kompendiet. Mens noen spørsmål krever at dere oppsøker informasjonen selv i dokumentasjon til Tensorflow eller andre sted på internett.

1. **Gitt følgende matriser**

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix}.$$

   (a) Utfør matrisemultiplikasjonen $\mathbf{C} = \mathbf{A} \cdot \mathbf{B}$. Husk at elementet $\mathbf{C}_{ij}$ beregnes ved å ta skalarproduktet av rad i i $\mathbf{A}$ og kolonne j i $\mathbf{B}$.

   (b) Forklar hvorfor dimensjonene til resultatmatrisen $\mathbf{C}$ blir 2×2.

   (c) Verifiser beregningene ved å finne hvert element i $\mathbf{C}$:

$$\mathbf{C} = \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix}.$$

1. Løsningsforslag

(a) Matrisemultiplikasjon $\mathbf{C} = \mathbf{A} \cdot \mathbf{B}$:

Gitte matriser er:

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix}.$$

Elementene i resultatmatrisen $\mathbf{C}$ beregnes ved:

$$c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj}.$$

Utførte beregninger:

$$c_{11} = 1 \cdot 1 + 2 \cdot 2 + 3 \cdot 3 = 1 + 4 + 9 = 14,$$

$$c_{12} = 1 \cdot 4 + 2 \cdot 5 + 3 \cdot 6 = 4 + 10 + 18 = 32,$$
$$c_{21} = 4 \cdot 1 + 5 \cdot 2 + 6 \cdot 3 = 4 + 10 + 18 = 32,$$
$$c_{22} = 4 \cdot 4 + 5 \cdot 5 + 6 \cdot 6 = 16 + 25 + 36 = 77.$$

Resultatet blir:
$$\mathbf{C} = \begin{bmatrix} 14 & 32 \\ 32 & 77 \end{bmatrix}.$$

Dette svarer også på oppgave 1c, hvor vi skal finne de fire elementene c_{11}, c_{12}, c_{21} og c_{22}. Matrisemultiplikasjon er nyttig i nevrale nettverk hvor vekter og biaser skal multipliseres med aktiveringen til ulike nevroner i nettverket.

(b) Forklar hvorfor dimensjonene til resultatmatrisen C blir 2×2:

Dimensjonen til en matrise etter multiplikasjon bestemmes av antall rader i $\mathbf{A}$ og antall kolonner i $\mathbf{B}$.

$\mathbf{A}$ har dimensjon 2×3, og $\mathbf{B}$ har dimensjon 3×2. Matrisemultiplikasjon er definert når antall kolonner i $\mathbf{A}$ er lik antall rader i $\mathbf{B}$, altså 3. Resultatet får dimensjonene 2×2 (rader i $\mathbf{A}$, kolonner i $\mathbf{B}$).

(c) Verifiser beregningene for matrisen C

Besvart over i løsningsforslag til 1a.

2. **Oppgave: Fra tabell til matrise og vektor**

   Tabellen nedenfor viser informasjon om fire personer, inkludert deres alder, antall år med utdanning, og om de jobber fulltid (ja = 1, nei = 0). Lønnen er oppgitt i tusen kroner.

Person	Alder	Utdanning (år)	Fulltid (1/0)	Lønn (Y, tusen kr)
1	25	16	1	450
2	30	14	0	350
3	40	18	1	600
4	35	12	0	400

   Svar på følgende oppgaver:

   (a) Skriv de uavhengige variablene som en matrise $\mathbf{X}$, der hver rad representerer en person, og hver kolonne representerer en av variablene.

(b) Skriv målvariabelen som en vektor $\mathbf{Y}$.

(c) Verifiser at dimensjonene til $\mathbf{X}$ og $\mathbf{Y}$ er henholdsvis 4×3 og 4×1.

Hint:

$$\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ \vdots & \vdots & \vdots \\ x_{n1} & x_{n2} & x_{n3} \end{bmatrix}, \quad \mathbf{Y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}.$$

2. Løsningsforslag

(a) Skriv de uavhengige variablene som en matrise X:

De naturlige uavhengige variablene i dette tilfelle er alder, utdanning (år), og fulltid (1/0). Matrisen $\mathbf{X}$ konstrueres ved å organisere disse variablene som følger:

$$\mathbf{X} = \begin{bmatrix} 25 & 16 & 1 \\ 30 & 14 & 0 \\ 40 & 18 & 1 \\ 35 & 12 & 0 \end{bmatrix}.$$

(b) Skriv målvariabelen som en vektor Y:

Målvariabelen $\mathbf{Y}$ er lønnen (Y), oppgitt i tusen kroner. Den representeres som en kolonnevektor:

$$\mathbf{Y} = \begin{bmatrix} 450 \\ 350 \\ 600 \\ 400 \end{bmatrix}.$$

(c) Verifiser dimensjonene til X og Y:

Dimensjonene til $\mathbf{X}$ er:

$$\mathbf{X} \text{ har 4 rader og 3 kolonner, altså dimensjonen } 4 \times 3.$$

Dimensjonene til $\mathbf{Y}$ er:

$$\mathbf{Y} \text{ har 4 rader og 1 kolonne, altså dimensjonen } 4 \times 1.$$

Dermed er dimensjonene til $\mathbf{X}$ og $\mathbf{Y}$ henholdsvis 4×3 og 4×1, som forventet.

3. **Tensorflow** I dette faget kommer vi til å bruke *Tensorflow*, som er et populært rammeverk for maskinlæring som støtter bruk av både CPU og GPU for å akselerere beregninger.

   Svar på følgende oppgaver:

   (a) Hva er en rank-0, rank-1, rank-2 og rank-3 tensor? Gjerne illustrer. Hva vil en rank-4 tensor være?

   (b) Tensorflow er optimalisert for å gjøre beregninger med matriser og tensorer. Hva er parallellbehandling? Hva er fordelen?

   (c) Forklar forskjellen mellom hvordan TensorFlow bruker CPU og GPU til beregninger. Hvorfor kan GPU være mer effektiv for dype nevrale nettverk sammenlignet med CPU?

   (d) Anta at du har installert TensorFlow på en maskin med både CPU og GPU. Hvordan kan du sjekke hvilke enheter TensorFlow bruker for å utføre beregningene? Gi et eksempel på en TensorFlow-kommando som lister tilgjengelige enheter.

3. Løsningsforslag

(a) Ulike tensorer

En tensors rank er definert som antall dimensjoner (eller akser) den har:

- **Rank-0 tensor:** En skalar (én enkelt verdi, uten dimensjoner). Eksempel: $x = 5$.
- **Rank-1 tensor:** En vektor (én dimensjon). Eksempel: $\mathbf{x} = \begin{bmatrix} 1 & 2 & 3 \end{bmatrix}$.
- **Rank-2 tensor:** En matrise (to dimensjoner: rader og kolonner). Eksempel: $\mathbf{X} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix}$.
- **Rank-3 tensor:** En 3D-tensor (tre dimensjoner, ofte brukt for RGB-bilder). Hvor den har en høyde, bredde og kanaler (eller dybde). Eksempel: $\mathbf{T}$ kan representeres som en "samling" av flere matriser.
- **Rank-4 tensor:** En 4D-tensor (fire dimensjoner, typisk brukt for batch-behandling i maskinlæring). Eksempel: En batch av bilder med form (batch size, høyde, bredde, kanaler).

(b) Parallellbehandling og TensorFlow

Parallellbehandling refererer til utførelsen av flere beregninger samtidig, i stedet for sekvensielt. I konteksten av TensorFlow, brukes flere kjerner i en CPU eller GPU for å fordele og akselerere beregningene.

Fordel: Parallellbehandling reduserer kjøretiden for store beregninger ved å utnytte flere kjerner samtidig, noe som er spesielt viktig for store matriseoperasjoner og trening av nevrale nettverk.

(c) CPU og GPU i TensorFlow

- **CPU:** CPU-en har færre kjerner med høy klokkefrekvens, som er optimalisert for sekvensielle oppgaver og generell databehandling.
- **GPU:** GPU-en har tusenvis av små kjerner designet for å håndtere store mengder data parallelt. Dette gjør GPU-en ideell for matriseoperasjoner og trening av dype nevrale nettverk, hvor mange beregninger skjer samtidig.

Hvorfor er GPU mer effektiv? Dype nevrale nettverk krever masse parallellbehandlinger, som GPU-er håndterer bedre enn CPU-er. For eksempel, i trening av et nevralt nettverk, kan GPU-en parallellisere vektoppdateringer og beregning av gradienter for alle noder/nevroner.

(d) Hvordan sjekke hvilke enheter TensorFlow bruker?

Du kan bruke følgende kommando for å liste opp enhetene som TensorFlow har tilgang til:

```
import tensorflow as tf
print(tf.config.list_physical_devices())
```

Denne kommandoen returnerer en liste over tilgjengelige enheter, for eksempel:

['CPU:0', 'GPU:0']

Eksempel: Hvis du ønsker å spesifisere hvilken enhet som skal brukes, kan du bruke følgende:

```
with tf.device('/GPU:0'):
    # Utfør operasjoner på GPU
    result = tf.matmul(a, b)
```

Vil du se hvor mange GPU'er du har, kan du skrive følgende:

```
import tensorflow as tf
print("Num GPUs Available: ", len(tf.config.list_physical_devices('GPU')))
```

Dette sørger for at operasjonen utføres på GPU-en. Når vi bruker Google Colab, kan vi velger hvilke kjøringstype (CPU/GPU) vi ønsker å kjøre koden på, og kan da velge f.eks. GPU.

4. **Normer i lineær algebra:** Gitt en vektor $\mathbf{v} = \begin{bmatrix} 3 \\ -4 \\ 1 \end{bmatrix}$:

   (a) Beregn ℓ_1-normen $\|\mathbf{v}\|_1 = \sum_{i=1}^{n} |v_i|$.

   (b) Beregn ℓ_2-normen $\|\mathbf{v}\|_2 = \sqrt{\sum_{i=1}^{n} v_i^2}$.

   (c) Sammenlign de to normene og diskuter hvordan ℓ_1- og ℓ_2-normer vektlegger elementene i vektoren.

4. Løsningsforslag

Gitt vektoren $\mathbf{v}$

(a) **Beregn ℓ_1-normen $\|\mathbf{v}\|_1$:**

$$\|\mathbf{v}\|_1 = \sum_{i=1}^{n} |v_i| = |3| + |-4| + |1| = 3 + 4 + 1 = 8.$$

(b) **Beregn ℓ_2-normen $\|\mathbf{v}\|_2$:**

$$\|\mathbf{v}\|_2 = \sqrt{\sum_{i=1}^{n} v_i^2} = \sqrt{3^2 + (-4)^2 + 1^2} = \sqrt{9 + 16 + 1} = \sqrt{26}.$$

$$\|\mathbf{v}\|_2 \approx 5.10 \quad \text{(avrundet til to desimaler).}$$

(c) **Sammenligning av ℓ_1- og ℓ_2-normene:**

- ℓ_1-normen summerer de absolutte verdiene av elementene i vektoren. Denne normen gir en direkte indikasjon på den totale "lengden" (eller størrelsen) av vektoren når vi ser på hver dimensjon individuelt.
- ℓ_2-normen beregner den euklidiske avstanden fra origo til vektoren i rommet. Den vektlegger større elementer mer, fordi den kvadrerer verdiene før summen beregnes. Dermed er ℓ_2-normen mer følsom for store elementer i vektoren.
- I dette tilfellet er ℓ_1-normen 8, mens ℓ_2-normen er omtrent 5.10. Generelt vil ℓ_1-normen være større enn eller lik ℓ_2-normen, ettersom ℓ_1-normen ikke kvadrerer elementene før summen beregnes. Dette kommer av noe som kalles $\ell_1\ell_2$-ulikheten (men det er ikke pensum her).

5. **Mean Squared Error (MSE) som en norm:** Gitt dataene $(\hat{y}_i, y_i)$ som representerer prediksjoner og sanne verdier for i-te datapunkt:

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^{n} (\hat{y}_i - y_i)^2.$$

(a) Forklar hvordan MSE kan relateres til ℓ_2-normen. Hint: Se på vektoren $\mathbf{r} = \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_n \end{bmatrix}$,

der $r_i = \hat{y}_i - y_i$, som residualvektoren.

Definisjon: Residualvektor

La $\mathbf{y}$ være en vektor som representerer de faktiske verdiene (observasjoner) og $\hat{\mathbf{y}}$ være en vektor med modellens prediksjoner. Da defineres residualvektoren $\mathbf{r}$ som:

$$\mathbf{r} = \mathbf{y} - \hat{\mathbf{y}},$$

hvor elementene i $\mathbf{r}$ er gitt ved:

$$r_i = y_i - \hat{y}_i, \quad \text{for } i = 1, 2, \ldots, n.$$

Tolkning:

- $r_i > 0$: Modellen undervurderte verdien (prediksjonen er lavere enn observasjonen).
- $r_i < 0$: Modellen overvurderte verdien (prediksjonen er høyere enn observasjonen).
- $r_i = 0$: Modellen predikerte nøyaktig riktig for dette datapunktet.

(b) Vis at MSE kan uttrykkes som:

$$\text{MSE} = \frac{1}{n}\|\mathbf{r}\|_2^2.$$

5. Løsningsforslag

Gitt dataene $(\hat{y}_i, y_i)$, hvor $\hat{y}_i$ er prediksjonen og y_i er den sanne verdien for i-te datapunkt.

(a) **Forklaring av hvordan MSE relateres til ℓ_2-normen:**
Residualvektoren $\mathbf{r}$ er definert som:

$$\mathbf{r} = \mathbf{y} - \hat{\mathbf{y}} = \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_n \end{bmatrix},$$

hvor hvert element $r_i = y_i - \hat{y}_i$ representerer avviket mellom den faktiske verdien og prediksjonen for datapunkt i.
Mean Squared Error (MSE) kan skrives som:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2.$$

Dette kan relateres til ℓ_2-normen ved å observere at:

$$\|\mathbf{r}\|_2^2 = \sum_{i=1}^{n} r_i^2 = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2.$$

Dermed kan MSE uttrykkes som:

$$\text{MSE} = \frac{1}{n}\|\mathbf{r}\|_2^2.$$

(b) **Vis at MSE $= \frac{1}{n}\|\mathbf{r}\|_2^2$:**
Dette har vi egentlig vist over i 5a, men men :-)
Fra definisjonen av residualvektoren $\mathbf{r}$, der $r_i = y_i - \hat{y}_i$, har vi:

$$\|\mathbf{r}\|_2^2 = \sum_{i=1}^{n} r_i^2 = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2.$$

Når vi deler summen av kvadrerte residualer på antall datapunkter n, får vi MSE:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \frac{1}{n}\|\mathbf{r}\|_2^2.$$

(c) **Tolkning:**
MSE er altså den gjennomsnittlige kvadrerte residualen. Dette betyr at MSE kvantifiserer modellens feil i kvadratiske termer, hvor større feil vektlegges mer enn mindre feil. Samtidig viser relasjonen til ℓ_2-normen (også kalt *euklidisk norm*) at MSE er en skalert versjon av residualvektorens lengde.

6. **Sammenligning mellom ℓ_p-normer og MSE:** Gitt residualvektoren $\mathbf{r} = \begin{bmatrix} -1 \\ 2 \\ -3 \\ 4 \end{bmatrix}$:

   (a) Beregn ℓ_1-normen $\|\mathbf{r}\|_1$, ℓ_2-normen $\|\mathbf{r}\|_2$, og ℓ_∞-normen $\|\mathbf{r}\|_\infty = \max |r_i|$.

   (b) Diskuter hvordan de forskjellige normene gir ulik informasjon om residualene.

   (c) Beregn MSE for residualvektoren, og forklar hvordan den er relatert til ℓ_2-normen.

7. **Praktisk anvendelse:** Du jobber med en regresjonsmodell som gir følgende prediksjoner:

$$\hat{\mathbf{y}} = \begin{bmatrix} 2.1 \\ 3.9 \\ 6.0 \\ 8.2 \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} 2 \\ 4 \\ 6 \\ 8 \end{bmatrix}.$$

   (a) Beregn residualvektoren $\mathbf{r} = \hat{\mathbf{y}} - \mathbf{y}$.

   (b) Finn $\|\mathbf{r}\|_1$, $\|\mathbf{r}\|_2$, og MSE.

   (c) Diskuter hvordan en lav $\|\mathbf{r}\|_1$ eller $\|\mathbf{r}\|_2$ kan være et mål for modellens kvalitet.