Mappeoppgave 2

Informasjon om oppgaven

Når du besvarer oppgaven, husk:

- les oppgaveteksten nøye
- kommenter koden din
- sett navn på akser og lignende i figurene
- skriv hvor du har hentet kodesnutter fra, hvis du gjør det
- bruk engelske variabelnavn og vær konsistent med hvordan du bruker store og små bokstaver
- bruk mest mulig funksjoner for ting som kan repeteres
- En kort kode kan være en bra kode, så ikke gjør det mer komplisert enn det spørres om.

Du kan få full pott uten å svare på oppgaven som er markert "ekstrapoeng". Du blir likevel belønnet for denne (dvs. hvis du har noen feil og får 45 poeng totalt, så kan du få en høyere poengsum hvis du også har svart på "ekstrapoeng".

Innlevering av oppgavene

Du skal levere begge mappene samtidig (det vil si denne oppgaven og mappe 1). Innleveringsfristen er 6 desember kl 13:00. Begge oppgavene skal leveres i github (som jupyter-fil) og wiseflow (som PDF). Bruk navnet "SOK-1003-eksamen-2022-mappe2" på filene.

- For github: Husk å gi meg (brukernavn "okaars") tilgang til github-reposetoriet deres. Hvis dere har satt reposetoriet til public (anbefales ikke), må dere dele lenken til dette på ole.k.aars@uit.no
- For wiseflow: En person fra hver gruppe (for hver mappeoppgave), leverer inn. Ved innlevering kan du krysse av hvem som er på gruppen din

Se generell informasjon om hvordan man leverer oppgaven her.

NB!: En person fra gruppa må fylle ut dette skjemaet for å melde om hvem som er på gruppa. Dere vil i etterkant motta en epost om tidspunkt for presentasjon.

Presentasjon

Presentasjonen innebærer en kort gjennomgang av oppgaven (10-15 min) etterfulgt av kommentarer fra meg (10-15 min). Alle gruppemedlemmer skal bidra til

presentasjonen. Det er anbefalt å laste opp besvarelsen på github forut for presentasjonen (helst to dager før) slik at jeg har mulighet til å lese gjennom. Dere vil ha mulighet til å endre besvarelsen etter presentasjonen, frem til endelig innlevering 6 desember.

Oppgave 1 (10 poeng)

a) Vi skal spille et spill der vi kaster en terning 6 ganger. Lag en funksjon med "for-løkke" som printer alle terningene som har blitt kastet. Du kan bruke `np.random.randint()` til å lage tilfeldige tall

In [129…
```
import random #impporterer pythons intigrerte random funksjon.

for x in range(6): #kaster terningen 6 ganger
    roll = random.randint(1, 6) #tilfeldige tall 1-6
    print(roll) #printer resultat
```

```
3
3
3
4
5
5
```

b) Juster den samme funksjonen slik at den lagrer tallene i en liste før den printer ut selve listen. Dere kan kalle denne listen for `lot_numbers`. Dere kan vurdere å bruke `append()` som del av funksjonen.

In [131…
```
import random #importerer pythons intigrerte random funksjon.

def dice(antall_ganger): #lager en funksjon for terningen
    lot_numbers = [] #lager en tom liste for resultatene
    for _ in range(antall_ganger): #kaster terningen x antall ganger
        roll = random.randint(1, 6) #lager resultatene med tall fra 1 og
        lot_numbers.append(roll) #legger kastene i listen resultat.
    return lot_numbers #returnerer listen med resultat

dice(6)
```

Out[131]:
```
[6, 1, 2, 6, 1, 1]
```

c) Juster den samme funksjonen slik at den har to argument. Disse argumentene er to terningverdier som du "tipper" blir kastet. Bruk `if`, `else` og `elif` til å generere vinnertall. Resultatet fra funksjonen skal printe ut ulike setninger avhengig av om man får 0, 1 eller 2 rette. Setningene velger du selv, men de skal inneholde tallene som du tippet, og tallene som ble trukket.

In [103…
```
import random #importerer pythons intigrerte random funksjon.

def gjett_dice(gjett1, gjett2): #lager en funksjon for ønsket formål
    lot_numbers = [] #lager en tom liste for resultatene
    for x in range(6): #kaster terningen x antall ganger
        roll = random.randint(1, 6) #lager resultatene med tall fra 1 og
        lot_numbers.append(roll) #legger kastene i listen resultat.
    rett_galt = ["Rett" if i == gjett1 else "Rett" if i == gjett2 else "G
    if rett_galt.count("Rett") == 0:
        print("Du gjettet " + str(gjett1)+ " og " + str(gjett2) + " og fi
    elif rett_galt.count("Rett") == 1:
        print("Du gjettet " + str(gjett1)+ " og " + str(gjett2) + " og fi
    elif rett_galt.count("Rett") == 2:
        print("Du gjettet tallene " + str(gjett1)+ " og " + str(gjett2) +
    else:
        print("Du gjettet tallene " + str(gjett1)+ " og " + str(gjett2) +
    return
#gjetter tallene 3 og 4
gjett_dice(3, 4)
```

```
Du gjettet 3 og 4 og fikk 1 rett av 6 kast. Du fikk følgende tall: [5, 1,
4, 1, 1, 1].
```

Oppgave 2 (10 poeng)

a) Du har nå begynt å spille lotto i stedet, og satser alt på ett vinnertall. Lag en while-løkke som printer ut tall helt til du har trukket riktig tall (som du definerer selv). For enkelthets skyld kan du begrense utfallsrommet av trekningene til mellom 0-30.

In [104…
```
import random

def lotto(eget_tall):
    lotto_tall = [] #lager liste med lotto tall
    while lotto_tall.count(eget_tall) == 0: #fortsetter loop til vi får v
        roll = random.randint(0, 30)  #random tall mellom 0 og 30
        lotto_tall.append(roll) #legger til i liste lotto_tall
    return lotto_tall

lotto(21)
```

Out[104]:
```
[15,
 25,
 25,
 3,
 4,
 20,
 7,
 8,
 18,
 8,
 0,
 13,
 9,
 22,
 9,
 2,
 30,
 30,
 16,
 3,
 9,
 8,
 2,
 25,
 5,
 0,
 24,
 28,
 5,
 14,
 5,
 6,
 12,
 21]
```

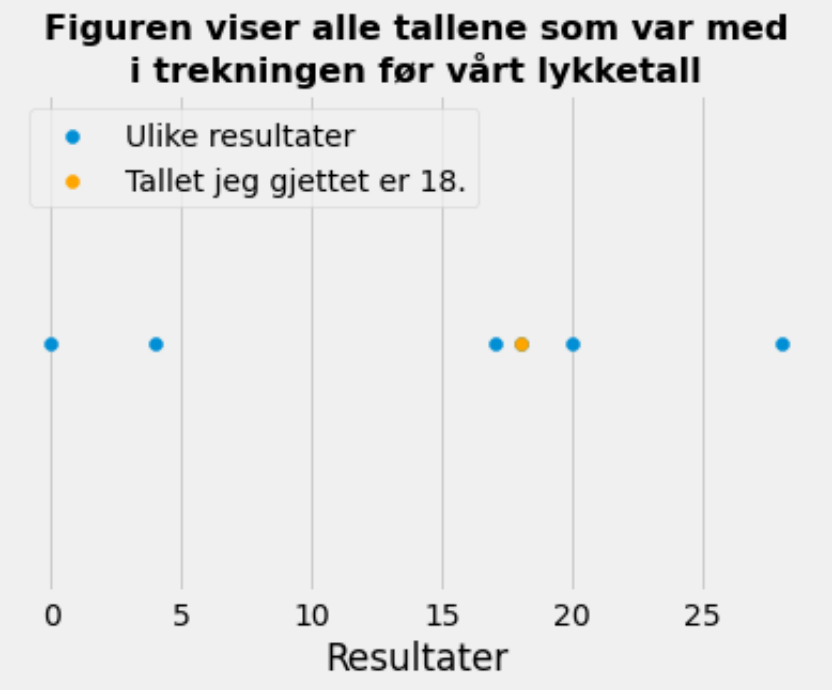
b) Lag et plot av den while-løkken du nettopp lagde. Man blir belønnet om man;

- bruker `scatter`;
- lager plottet dynamisk (dvs at hver trekning vises hver for seg, og at x-aksen endrer seg etter en gitt verdi);
- viser hvor når siste trekningen blir gjort (dvs at den vises kun når du har trukket vinnertallet).

Avhengig av hvordan du lager figuren din kan du får bruk for å importere pakkene `Ellipse`, `display`, `clear_output`.

In [105…
```
#importerer pyplot
import matplotlib.pyplot as plt
from matplotlib import style #pakke for tema i pyplot

#tall man ønsker å gjette
tall_jeg_gjetter = 18

#definerer x til bruk i histogram
x1 = lotto(tall_jeg_gjetter)

#lager liste uten duplicates for bruk i scatter.
uten_duplicates = [] #tom liste
for i in x1: #for funksjon som legger til alle tall fra listen lotto(tall
    if i not in uten_duplicates:
        uten_duplicates.append(i)

plt.style.use("fivethirtyeight") #tema
plt.scatter(uten_duplicates, [5]*len(uten_duplicates), label='Ulike resul
plt.scatter(tall_jeg_gjetter, [5]*len([tall_jeg_gjetter]), color="orange"
plt.yticks([]) #fjerner ylabels da disse er urelevante
plt.legend(loc='upper left',frameon=True) #legendeposisjon
plt.title("""Figuren viser alle tallene som var med
i trekningen før vårt lykketall""", weight="bold", fontsize=16) #figurtit
plt.xlabel('Resultater') #x akse tittel
```

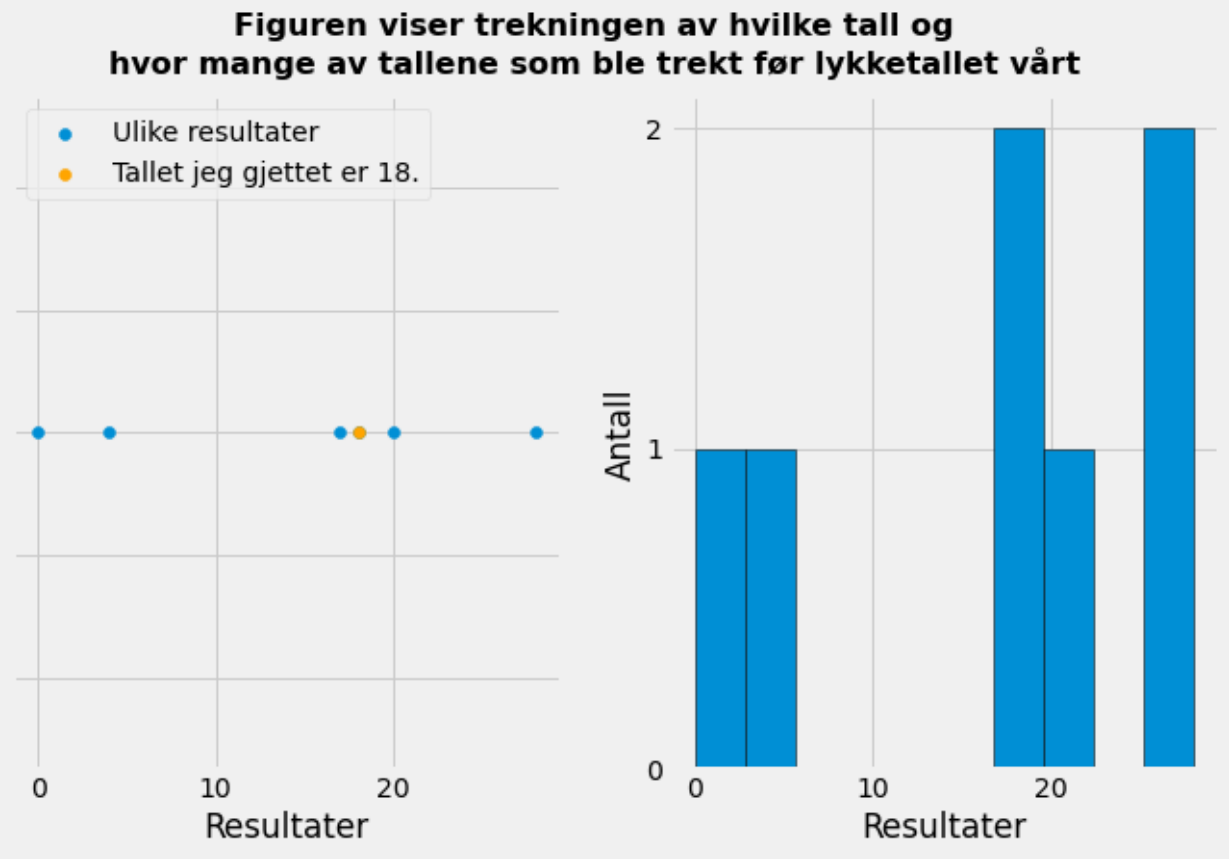
Out[105]:
```
Text(0.5, 0, 'Resultater')
```



c) Ekstrapoeng: gjør det samme som i (b), men lag et histogram som vises ved siden av. Dette histogrammet skal vise hvor mange ganger de ulike tallene ble trekt. Bruk `plt.hist` til dette. Husk at du må definere figur og akseobjekt først.

In [106…
```
from matplotlib.ticker import MaxNLocator #pakke for å gjøre yaxis bare i
from matplotlib.cbook import get_sample_data

#lager figur fig
fig,(ax1, ax2)=plt.subplots(1,2, figsize=(10,6)) #definerer fig
fig = plt.style.use("fivethirtyeight") #tema
fig = plt.suptitle("""Figuren viser trekningen av hvilke tall og
hvor mange av tallene som ble trekt før lykketallet vårt""", fontsize=16,

#definerer figurene

ax1.scatter(uten_duplicates, [5]*len(uten_duplicates),label='Ulike result
ax1.scatter(tall_jeg_gjetter, [5]*len([tall_jeg_gjetter]),label='Tallet j
ax2.hist(x1, edgecolor="black") #histogram med svart rundt kantene
ax2.yaxis.set_major_locator(MaxNLocator(integer=True)) #setter y axis på
ax1.legend(loc='upper left',frameon=True) #legende posisjon
ax1.set_yticklabels([]) #fjerner y labels ax1.
ax1.set_xlabel('Resultater') #x akse tittel ax1
ax2.set_xlabel('Resultater') #x akse tittel ax2
ax2.set_ylabel('Antall') #y akse tittel ax2
```

Out[106]:
```
Text(0, 0.5, 'Antall')
```



Oppgave 3 (20 poeng)

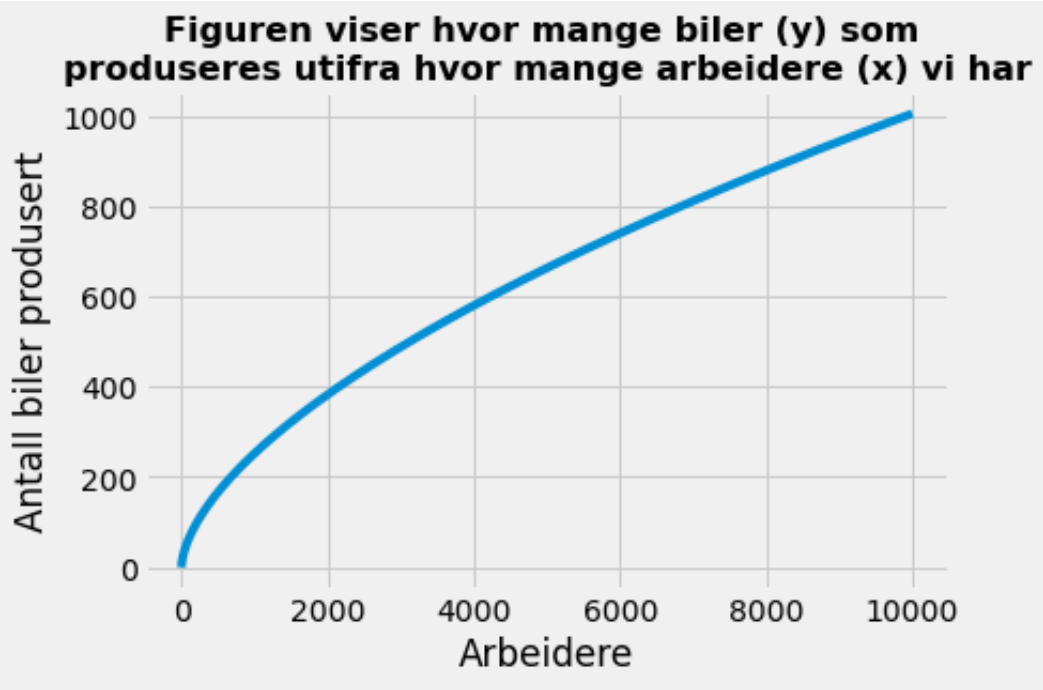
En bedrift produserer biler. Produktfunksjonen til bedriften defineres slik $f(L, a, R) = 2RL^{a}$, hvor:

- `L` er arbeidskraft,
- `a` er produktiviteten til arbeiderne og
- `R` er antall robotmaskiner

a) Lag en formel for produktfunksjonen til bedriften og plot den grafisk med ulike verdier av `L` på x-aksen. Anta `a=0.6` og `R=2`

In [107…
```
import numpy as np #importerer numpy as np

x = np.arange(0, 10000, 1, dtype=int) #lager liste med heltall fra 0 til
y = 2*2*x**0.6 #setter inn tallene i formelen for funksjonen

plt.plot(x,y)
plt.title("""Figuren viser hvor mange biler (y) som
produseres utifra hvor mange arbeidere (x) vi har""", weight="bold", font
plt.xlabel("Arbeidere") #x akse tittel
plt.ylabel("Antall biler produsert") #y akse tittel
```

Out[107]:
```
Text(0, 0.5, 'Antall biler produsert')
```



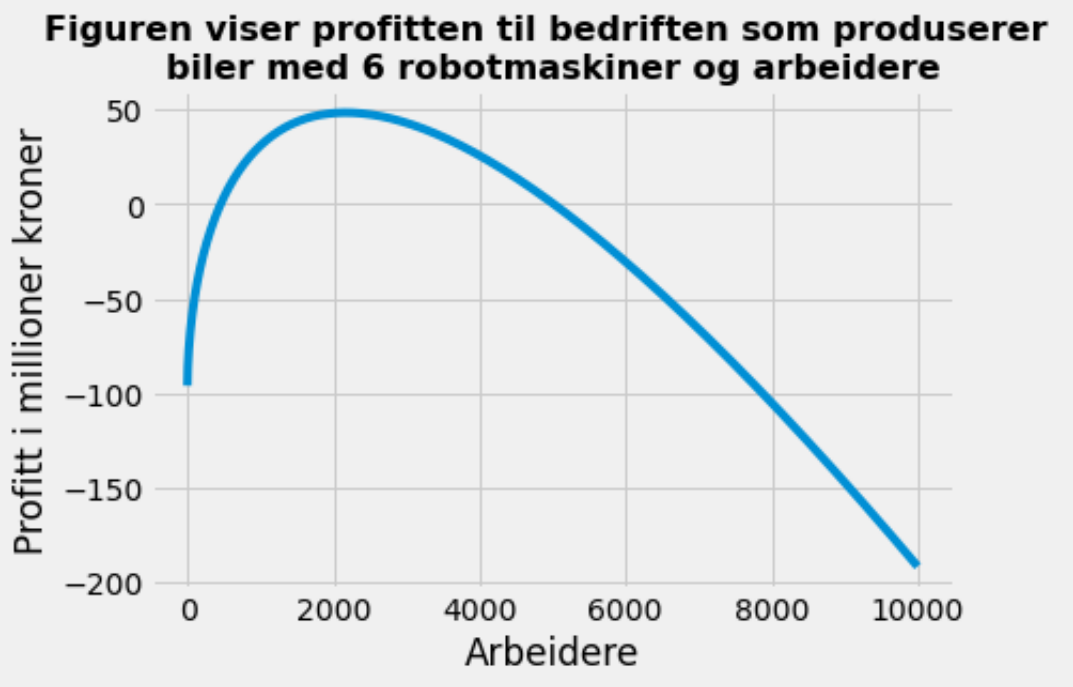
b) anta at profittfunksjonen til denne bedriften er ${profit = f(L, a, R)p-wL-cR-K}$, hvor

- `w` er månedslønnen til arbeiderne,
- `c` er kostnaden for robotmaskinene
- `K` er faste kostnader
- `p` er utsalgsprisen på bilene.

Anta `a=0.6`, `R=6`, `p=300 000`, `w=100 000`, `c= 1 000 000` og `K=90 000 000`. Plot profittfunksjonen figurativt for antall arbeidere (`L`) mellom 0 og 10 000. Vis profitten i millioner (dvs at du må dele på 1 000 000)

In [108…
```
#bruker samme x fra forrige oppgave, da denne er en liste med x verdier f

y2 = (((2*6*x**0.6)*0.3)-(0.1*x)-(1*6)-90)

plt.plot(x,y2)
plt.title("""Figuren viser profitten til bedriften som produserer
biler med 6 robotmaskiner og arbeidere""", weight="bold", fontsize=16) #t
plt.xlabel("Arbeidere") #x akse tittel
plt.ylabel("Profitt i millioner kroner") #y akse tittel
```

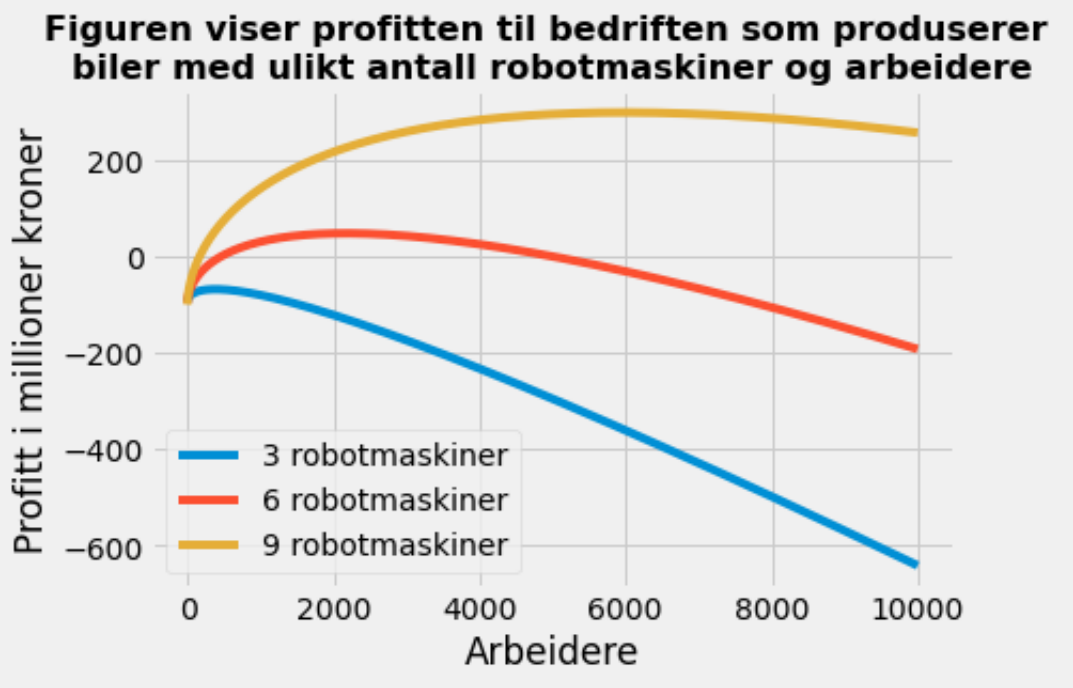
Out[108]:
```
Text(0, 0.5, 'Profitt i millioner kroner')
```



c) Plot profitttfunksjonen for antall robostmaskiner `R=[3, 6, 9]` i samme plot (dvs at tre profittfunksjoner vises sammen). Bruk av "for loops" for å gjøre dette belønnes

In [109…
```
#bruker x verdi fra oppg 3a
y3 = ((2*3*x**0.6)*0.3)-(0.1*x)-(1*3)-90 #3 robotmaskiner y verdi
y6 = ((2*6*x**0.6)*0.3)-(0.1*x)-(1*6)-90 #6 robotmaskiner y verdi
y9 = ((2*9*x**0.6)*0.3)-(0.1*x)-(1*9)-90 #9 robotmaskiner y verdi

plt.plot(x,y3, label="3 robotmaskiner") #3 robotmaskienr plot
plt.plot(x,y6, label="6 robotmaskiner") #6 robotmaskiner plot
plt.plot(x,y9, label="9 robotmaskiner") #9 robotmaskiner plot
plt.title("""Figuren viser profitten til bedriften som produserer
biler med ulikt antall robotmaskiner og arbeidere""", weight="bold", font
plt.xlabel("Arbeidere") #x akse tittel
plt.ylabel("Profitt i millioner kroner") #y akse tittel
plt.legend(loc='lower left',frameon=True) #legendeposisjon
```

Out[109]: `<matplotlib.legend.Legend at 0x7f1ba4f12670>`



d) finn profittmaksimum og optimal antall arbeidere ved hjelp av derivasjon med samme forutsetninger som i (1b). Bruk `sympy` -pakken til dette

In [110…
```
#pakker du kan få bruk for
import sympy as sp
from sympy.solvers import solve


u = sp.symbols("x") #definerer u som x variabelen
z = sp.solve(sp.Eq(sp.diff((((2*6*u**0.6)*0.3)-(0.1*u)-(1*6)-90)), 0), u)

print("Antall arbeidere som gir oss mest profitt er " + str(int(z[0]))) #
```

```
Antall arbeidere som gir oss mest profitt er 2168
```

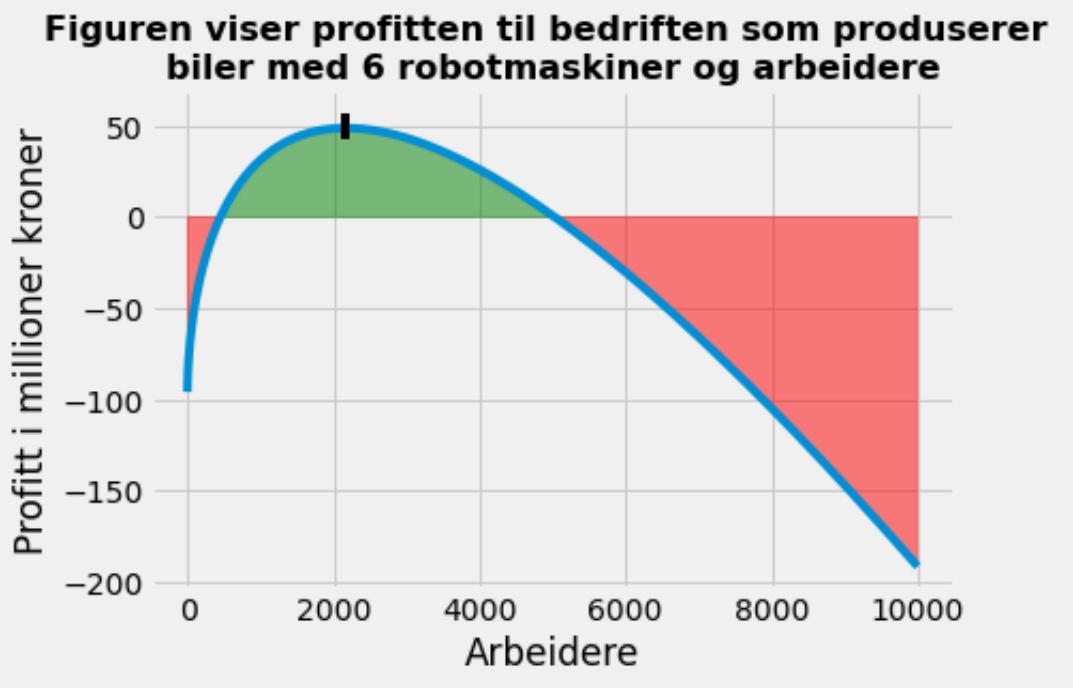
e) vis figurativt med bruk av `fill_between` arealet hvor man taper penger (i rødt) og hvor man tjener penger (i grønt). Marker også profittmaksimum og antall arbeidere i profittmaksimum - gjerne ved bruk av `vlines`. Bruk ellers samme forutsetninger for argumentene som i oppgave (1b)

In [111…

```
#bruker samme x fra oppgave 3a, da denne er en liste med x verdier fra 0

plt.plot(x,y2) #plot
plt.title("""Figuren viser profitten til bedriften som produserer
biler med 6 robotmaskiner og arbeidere""", weight="bold", fontsize=16)
plt.xlabel("Arbeidere") #x akse tittel
plt.ylabel("Profitt i millioner kroner") #y akse tittel
plt.vlines(x = 2168, ymin = 43, ymax = 57, color ='black', label="Inntekt
plt.fill_between(x, y2, where = (y2 > 0), alpha=0.5, color = 'g') #PROFIT
plt.fill_between(x, y2, where = (y2 < 0), alpha=0.5, color = 'r') #LOSS f
```

Out[111]:

```
<matplotlib.collections.PolyCollection at 0x7f1ba4eba730>
```



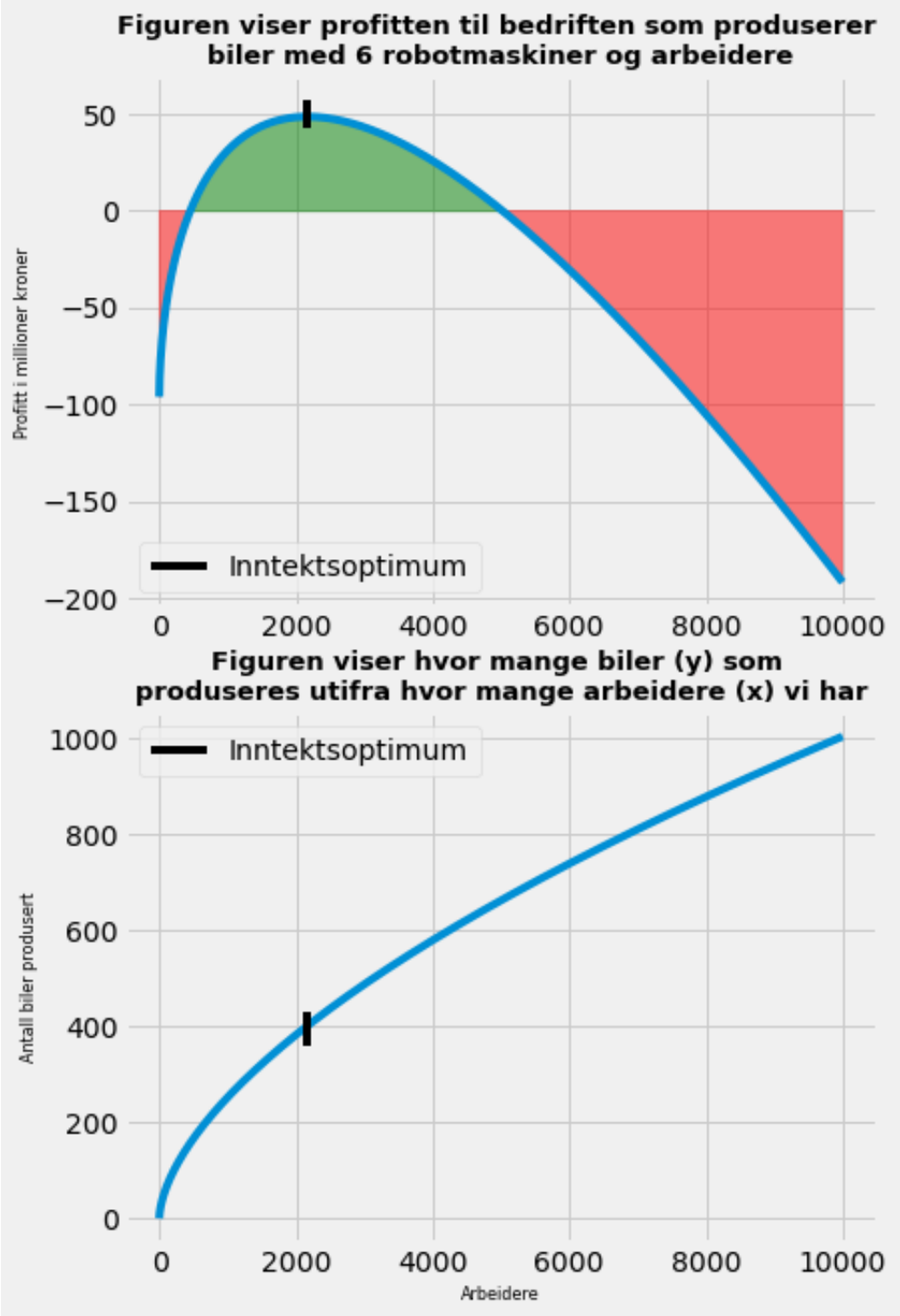
f) Plot nå to figurer sammen der du viser hva optimal antall arbeidere gir i profitt (slik som i (2e)) og produksjon av antall biler (som du får fra produktfunksjonen). Marker optimum med vlines. Ha grafen med profittfunksjonen over grafen med produktfunksjonen. Du kan bruke `fig, (ax1, ax2) = plt.subplots(2)` når du skal gjøre dette.

Hint: Du kan finne antall biler som blir produsert ved å bruke antall arbeidere i profittmaksimum, i produktfunksjonen.

In [112…

```
fig2, (ax3, ax4) = plt.subplots(2, figsize=(6,10)) #definerer fig2 som 2

ax3.plot(x,y2) #plot
ax3.set_title("""Figuren viser profitten til bedriften som produserer
biler med 6 robotmaskiner og arbeidere""", weight="bold", fontsize=13) #t
ax3.set_ylabel("Profitt i millioner kroner", fontsize=8) #y akse tittel
ax3.fill_between(x, y2, where = (y2 > 0), alpha=0.5, color = 'g') #PROFIT
ax3.fill_between(x, y2, where = (y2 < 0), alpha=0.5, color = 'r') #LOSS f
ax3.vlines(x = 2168, ymin = 43, ymax = 57, color ='black', label="Inntekt
ax3.legend(loc='lower left',frameon=True) #legendeposisjon
ax4.plot(x,y) #plot
ax4.set_title("""Figuren viser hvor mange biler (y) som
produseres utifra hvor mange arbeidere (x) vi har""", weight="bold", font
ax4.set_xlabel("Arbeidere", fontsize=8) #x akse tittel
ax4.set_ylabel("Antall biler produsert",fontsize=8) #y akse tittel
ax4.vlines(x = 2168, ymin = 360, ymax = 430, color ='black', label="Innte
ax4.legend(loc='upper left',frameon=True) #legendeposisjon
```

Out[112]: <matplotlib.legend.Legend at 0x7f1ba4de3b20>



Oppgave 4 (10 poeng)

I denne oppgaven skal vi hente ut et datasett fra eurostat på investeringer i hosholdningen. Bruk koden under til å hente ut dataene.
NB!: Husk at dere må ha innstallert pakken `eurostat`. Dette gjør dere med å åpne "Terminal" og kjøre `pip install eurostat`.

In [113…

```
import eurostat
import pandas as pd

inv_data = eurostat.get_data_df('tec00098') #laster in datasett
```

a) Bytt navn på kolonnen "geo\time" til "country" ved bruk av en av kodene under. Fjern så alle kolonner utenom "country" og alle årstallene.
NB!: Noen vil få en ekstra første kolonne som heter "freq" eller noe annet. Da må dere bruke versjon 2 av koden under.

In [114…
```
inv_data.columns = ['freq', 'unit',   'sector',  'na_item', 'country'] +
```

In [115…
```
chosen_columns=['country']+list(range(2010,2022))
inv_data=inv_data.loc[:,chosen_columns] #lagrer bare de kolonnene jeg øns
```

b) fjern radene med nan verdi. Sett deretter indeksen til "country".
Hint: En metode er å bruke `isna()` og `any()` over radaksene (dvs. `axis=1`)

In [116…
```
inv_data.dropna(inplace=True) #dropper rader med na
inv_data.set_index('country') #setter country som index
```

Out[116]:

country	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	2
AT	8.44	8.60	8.47	8.58	8.30	8.36	8.29	8.69	8.81	9.03	9.21	9
BE	9.82	9.45	9.37	9.03	9.45	9.31	9.33	9.28	9.35	9.78	9.15	9
CY	13.72	10.73	8.74	7.53	7.15	6.69	8.02	8.97	11.30	12.98	13.12	13
CZ	11.14	9.82	8.74	8.77	8.88	8.84	9.18	7.86	9.00	9.45	9.34	9
DE	8.72	9.51	9.61	9.56	9.65	9.32	9.62	9.48	9.68	9.71	9.95	10
DK	8.50	8.51	7.87	7.35	7.60	7.50	7.46	8.10	8.33	8.57	8.69	9
EA19	9.30	9.21	8.80	8.37	8.24	8.09	8.35	8.52	8.71	8.77	8.53	
EE	6.14	6.47	6.84	7.52	7.79	7.94	8.52	8.99	8.68	9.26	9.94	10
EL	9.12	8.54	6.28	4.90	3.16	2.72	2.75	2.69	2.44	2.59	2.97	3
ES	9.66	7.85	6.48	4.84	4.72	4.57	4.60	5.17	5.42	5.54	5.34	6
EU27_2020	9.07	8.95	8.56	8.12	8.06	7.97	8.22	8.42	8.54	8.60	8.33	
FI	11.66	12.18	12.12	11.49	10.88	10.49	11.57	12.09	12.50	12.25	11.91	12
FR	9.31	9.46	9.31	9.27	9.07	8.88	8.99	9.41	9.47	9.62	8.57	9
HR	5.96	5.66	5.53	5.01	4.78	4.91	4.96	5.01	5.35	6.37	6.40	6
HU	6.67	5.11	4.80	4.92	5.17	5.68	5.86	6.53	7.19	7.52	8.87	8
IE	6.01	4.96	4.12	4.29	4.41	4.72	5.26	5.50	6.27	4.92	4.27	5
IT	10.33	9.82	9.20	8.42	7.78	7.58	7.68	7.74	7.77	7.64	7.20	8
LT	4.73	5.11	4.70	5.56	6.04	6.68	6.89	6.51	6.78	7.09	7.04	
LU	9.69	11.26	10.43	11.34	12.75	12.34	12.48	12.35	11.45	10.34	9.71	1
LV	3.56	4.92	6.17	4.62	4.98	5.76	4.87	4.83	5.68	5.85	5.30	4
NL	10.05	9.54	8.47	7.46	8.10	9.08	10.60	10.80	11.52	12.15	12.22	12
NO	9.71	10.98	11.74	12.32	11.82	11.35	12.34	13.02	12.22	11.89	11.29	
PL	7.99	7.92	8.24	7.59	7.86	7.88	7.52	6.88	5.84	5.93	5.18	6
PT	6.27	5.81	5.07	4.37	4.51	4.50	4.71	5.05	5.48	5.65	5.69	
SE	5.86	5.48	4.51	4.61	4.69	5.69	6.15	6.77	6.24	5.88	6.48	6
SI	6.92	6.30	5.78	5.50	5.65	5.81	5.85	6.28	6.51	6.33	5.64	6
SK	6.32	7.00	6.65	7.08	6.39	6.10	6.75	6.71	6.76	6.79	6.93	

c) Lag et nytt datasett hvor du kun har med de nordiske landene (dvs. "NO", "SE", "DK", "FI"). Det kan være nyttig å bruke `isin` til dette. Bytt så om på kolonner og rader ved hjelp av `transpose`.

In [117…
```
wanted=inv_data['country'].isin(['NO','SE','DK','FI']) #lager liste med d

inv_nordic = inv_data[wanted==True] #lager datasett med bare ønskede lang

inv_nordic.set_index("country") #setter country som index

inv_nordic=inv_nordic.transpose() #transpose

inv_nordic = inv_nordic.rename(columns=inv_nordic.iloc[0]) #setter column

inv_nordic =inv_nordic.iloc[1: , :] #fjerner første rad da disse er navne

inv_nordic
```

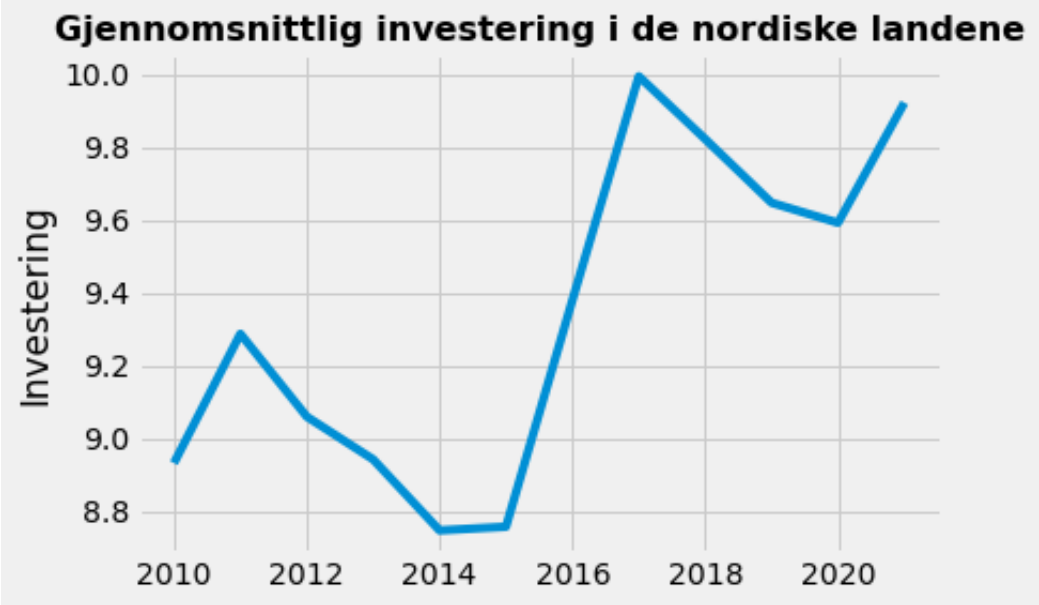
Out[117]:

	DK	FI	NO	SE
2010	8.5	11.66	9.71	5.86
2011	8.51	12.18	10.98	5.48
2012	7.87	12.12	11.74	4.51
2013	7.35	11.49	12.32	4.61
2014	7.6	10.88	11.82	4.69
2015	7.5	10.49	11.35	5.69
2016	7.46	11.57	12.34	6.15
2017	8.1	12.09	13.02	6.77
2018	8.33	12.5	12.22	6.24
2019	8.57	12.25	11.89	5.88
2020	8.69	11.91	11.29	6.48
2021	9.24	12.5	11.11	6.84

d) Lag en ny kolonne som du kaller "mean". Denne skal være gjennomsnittet av alle de nordiske landene for hvert av årene (dvs at du må ta gjennomsnittet over radene). Plot så dette og kall y-aksen for "investering"

In [118…
```
inv_nordic["mean"]=inv_nordic.mean(axis=1) #lager en gjennomsnittlig invi

plt.plot(inv_nordic["mean"]) #plotter gjennomnsnittet
plt.ylabel("Investering") #y akse tittel
plt.title("""Gjennomsnittlig investering i de nordiske landene""", weight
```

Out[118]:
```
Text(0.5, 1.0, 'Gjennomsnittlig investering i de nordiske landene')
```


Gjennomsnittlig investering i de nordiske landene
Investering
10.0
9.8
9.6
9.4
9.2
9.0
8.8
2010
2012
2014
2016
2018
2020