Introduksjon til Dataanalyse med Python

Geir Arne Hjelle / Tekna

5. mars 2024

Velkommen til kurset **Introduksjon til Dataanalyse med Python**. Dette dokumentet beskriver hvordan du kan installere Python på din maskin, forteller litt om bakgrunnen for at Python har blitt et veldig populært programmeringsspråk, og gir deg noen små raske oppgaver.

Oppgavene er hovedsaklig ment som en sjekk på at installasjonen har fungert, samt å gi deg en liten forsmak på noen av mulighetene har å by på. Vær snill og installer Python samt jobb gjennom oppgavene **før kurset starter**. Dermed kan vi bruke kurstiden på å lære Python. Om du har noen problemer med installasjonen eller spørsmål til oppgavene så **send en e-post til kursleder Geir Arne:** `geirarne@gmail.com`

Installer Python

På python.org vil du alltid finne siste versjon av Python. Det er likevel enklere å installere en Python-distribusjon kalt Anaconda. Anaconda inneholder flere nyttige tilleggspakker og programmer:

- Python
- De mest brukte Pythonpakkene
- Conda - En pakkebehandler som brukes for å installere flere støttepakker
- Jupyter - Et miljø som kjører i nettleseren, og som er effektivt for interaktiv programmering og utforskning
- Flere editorer tilpasset Python for å skrive programmer

Anaconda er tilgjengelig for både Windows, Mac og Linux, og kan installeres uten administrative rettigheter.

- Last ned og installer **Anaconda** fra www.anaconda.com/products/individual. Du kan finne detaljerte instruksjoner for installasjon på hjemmesiden til Anaconda: docs.anaconda.com/anaconda/install/windows

  *Python er allerede installert på Mac og Linux. Du bør **ikke** bruke disse preinstallerte versjonene av Python, fordi de brukes av systemet. Det gjør det vanskelig å oppdatere til nyere versjoner, og om noe skulle gå galt er det fare for at du ødelegger for hele operativsystemet. Bruk Anaconda også på Mac og Linux.*

Etter at du har installert Anaconda er programmet **Anaconda Navigator** tilgjengelig. Du kan bruke dette programmet til å starte forskjellige programmeringsverktøy, inkludert Pythonterminaler, Jupyter og editorer.

Anaconda er en ganske tung installasjon. Om du ønsker en mindre installasjon, hvor du bare installerer pakker og verktøy ved behov, kan du installere Miniconda i stedet. Dette krever dog at du er komfortabel med å jobbe med kommandolinjen.

Oppgave 1: Hei Python

Med Python installert er det på tide å se hvordan man bruker det. La oss skrive vårt første program:

- Start **Anaconda Navigator**
- Klikk på **Environments**
- Klikk på ▷ til høyre for **Base**
- Velg **Open Terminal**

Dette åpner en terminal hvor Anaconda Python er tilgjengelig.

- Skriv `python` og trykk **Enter**

Du kommer nå inn i et miljø kalt **Python REPL**. Du vil se noe tekst og nederst vil du se >>>, som kalles Python-promptet. Her kan du skrive Pythonkommandoer og de vil bli utført umiddelbart.

- Skriv inn følgende kode (Tallet `1` er et linjenummer og ikke en del av koden):

Python

```
1 print("Hei Python!")
```

Terminalen skal svare med å skrive ut teksten `Hei Python!`. Hvis den ikke gjør det, men du i stedet får en feilmelding så sjekk nøye at du har skrevet alt riktig og prøv igjen. Det er spesielt viktig å stave `print` riktig, samt å bruke parantesene og fnuttene som beskrevet.

> **Merk:** I denne og andre kodesnutter i dette heftet vil du se linjenummer til venstre for koden. Disse skal du ikke skrive inn. De er bare brukt for å markere og kunne referere til enkeltlinjer i kodesnuttene.

Når terminalen svarer `Hei Python!` har du skrevet og kjørt en Pythonkommando! Dette bekrefter også at installasjonen av Python fungerte.

Kort forklaring av koden:

- `print()` er en kommando som brukes for å skrive ting til terminalvinduet. Navnet på kommandoen henger igjen fra gamle dager da man ikke hadde skjerm, men måtte fysisk skrive ut resultater for å se dem.
- `"Hei Python!"` er en tekststreng. Python bruker fnutter (enten enkle, `'`, eller doble, `"`) for å markere tekststrenger. Disse blir ikke tolket av Python slik at du kan skrive (nesten) hva som helst mellom fnuttene.

Trykk *pil opp*. Dette henter tilbake den forrige kommandoen du skrev. Endre tekststrengen til noe annet, for eksempel navnet ditt. Kjør kommandoen en gang til.

Interaktiv Programmering i REPL

REPL står for **Read-Eval-Print-Loop**. Denne er veldig nyttig for å teste ut små kodesnutter interaktivt, men kan ikke brukes til å skrive fullstendige programmer som skal brukes flere ganger.

- Skriv inn følgende kode i REPL. Du må først skrive linje 1 og trykke *Enter*. Deretter kan du skrive inn linje 2 og se resultatet. Bruk gjerne ditt eget navn i stedet for `"Geir Arne"`:

Python

```
1 navn = "Geir Arne"
2 print(f"Hei {navn}")
```

Kort forklaring av koden:

- `"Geir Arne"` er en tekststreng, akkurat som `"Hei Python!"` var en tekststreng tidligere.

- `navn` er en variabel. Du bruker `=` for å tilordne verdier til variabler som du kan referere til senere. Her sier du at navnet (`"Geir Arne"`) skal refereres til som `navn`. Du kan bruke nesten hvilke ord som helst som variabler i Python men variabelnavn kan ikke inneholde mellomrom.
- `print()` kjenner du allerede, den skriver noe til skjermen.
- `f"Hei {navn}"` er en spesiell tekststreng. `f` som står foran tekststrengen markerer den som en **formattert tekst**, ofte kalt en **f-string** på engelsk. Disse er nyttige fordi du kan referere til variabler ved å skrive de mellom krøllparanteser. Legg merke til at `{navn}` byttes ut med svaret du skrev på spørsmålet `Hva heter du?`

Som tidligere, sjekk at du har skrevet ting riktig om det dukker opp feilmeldinger. Eksperimenter ved å endre litt i koden. Hva skjer om du tar bort `f` i `print()` kommandoen: `print("Hei {navn}")`? Hva skjer om du skriver `navn` uten krøllparanteser: `print(f"Hei navn")`

En forskjell mellom REPL og vanlige Pythonskript (som du vil se senere) er at verdien av uttrykk automatisk skrives til skjermen:

Python

```
1 navn
```

Dette vil skrive ut teksten `navn` refererer til. Legg merke til at REPL skriver ut fnutter for å markere at `navn` er en tekststreng. `print()` skriver ikke ut de samme fnuttene.

Du kan også bruke Python som en kalkulator med tegnene `+` (pluss), `-` (minus), `*` (gange), `/` (dele) og `**` (opphøyd i):

Python

```
1 (6048 + 1729) ** 2
```

> Avslutt den kjørende REPL-sessjonen ved å skrive `exit()`. Du kan også bruke enten *Ctrl-Z* eller *Ctrl-D* avhengig av operativsystemet ditt.

Oppgave 2: Ditt første program

Det kan være nyttig å eksperimentere ved å skrive kode i REPL-miljøet. Men det er oftest mer nyttig å forberede programmer som lett kan kjøres om og om igjen.

Et Python-program er en ren tekstfil som inneholder Pythonkommandoer. For å skrive slike programmer kan du bruke hvilken som helst teksteditor (for eksempel Notisblokk),

men **ikke** tekstbehandlere (for eksempel Word) som formatterer teksten. I denne oppgaven skal du skrive og kjøre et Pythonprogram.

- Start **Anaconda Navigator**
- Finn programmet **Spyder** og klikk **Launch**. Om Spyder ikke er installert må du først klikke **Install**

Dette åpner teksteditoren **Spyder**. Her kan du skrive Pythonprogrammer. Den venstre siden av Spyder består av et vindu hvor du kan skrive din egen kode.

Slett kommentarene som allerede står der, og skriv inn den samme koden du skrev tidligere:

Python

```python
navn = "Geir Arne"
print(f"Hei {navn}")
```

- Lagre programmet ditt ved å velge *File* > *Save as ...* i menyen.
- Finn en passende katalog hvor du finner igjen programmet senere. Gi programmet ditt navnet `heisann.py`. **NB:** Du bør lagre programmet på en lokal disk. Noen har opplevd små forskjeller videre i kurset om programmet er lagret for eksempel på en Sharepoint-disk.

Pythonprogrammer bruker endelsen `.py`. Du kan velge ganske fritt hvordan du navngir programmene dine, men det er lurt å unngå mellomrom (bruk understrek, _, i stedet).

- I Spyder kan du kjøre programmet ditt ved å trykke på den grønne ▷ på menylinjen. Du kan også trykke *F5* eller velge *Run* > *Run* i menyen.
- Du vil nå se hilsenen i vinduet nede til høyre. Legg merke til at begge kommandoene ble kjørt direkte etterhverandre.

Når du kjører et program kjører alle kommandoene i programmet etter hverandre.

- Prøv å legge til flere kommandoer.
- Om du bare legger til et uttrykk, for eksempel en linje som kun sier `(6048 + 1729) ** 2`, vil ikke denne skrives ut av programmet ditt, selv om den ble skrevet ut i REPL-sesjonen. Du må bruke `print()` for å vise verdien av uttrykk: `print((6048 + 1729) ** 2)`.

Oppgave 3: Les en Excelfil

Programmer er mer interessante når de brukes til å behandle virkelige data. I dette kurset skal vi se spesielt på hvordan kan gjøre enkel dataanalyse. Som en enkel introduksjon skal du lese inn en Excelfil.

- Gå til nettsiden www.regjeringen.no/no/statsbudsjett/2020/rnb og last ned regnearket `kap1.xlsx` ved å klikke **Tallene bak figurene** og deretter Excelarket som er lenket til **Kapittel 1**. Lagre det i samme katalog som der du lagret Pythonprogrammet ditt tidligere.

For å jobbe med dataanalyse i Python skal vi hovedsaklig bruke en pakke som heter **pandas**. pandas er ikke en del av standardinstallasjonen av Python, men ble installert av Anaconda.

- Start et nytt program i **Spyder** ved å trykke *Ctrl-N* eller velg *New File* fra menyen eller menylinjen.
- Lagre filen som `budsjett.py` i samme katalog som du lagret filen `kap1.xlsx`. Ta gjerne en titt på filen for å se på innholdet.
- Skriv inn følgende kode:

Python

```python
import pandas as pd

# Les filen kap1.xlsx
budsjett = pd.read_excel(
    "kap1.xlsx",
    sheet_name="1.2",
    header=4,
    index_col=0,
    usecols=[0, 1, 2],
    na_values="-",
)

print(budsjett)

# Tegn dataene i en figur
budsjett.plot.bar(stacked=True)
```

Kort forklaring av koden:

- `import pandas as pd`: `import` er en kommando for å laste pakker inn i programmet vårt. Når vi vil bruke pandas må vi fortelle Python det ved å importere den. Ved å skrive `as pd` sier vi at vi vil referere til pandas som `pd`. Dette er en vanlig forkortelse når man jobber med pandas.

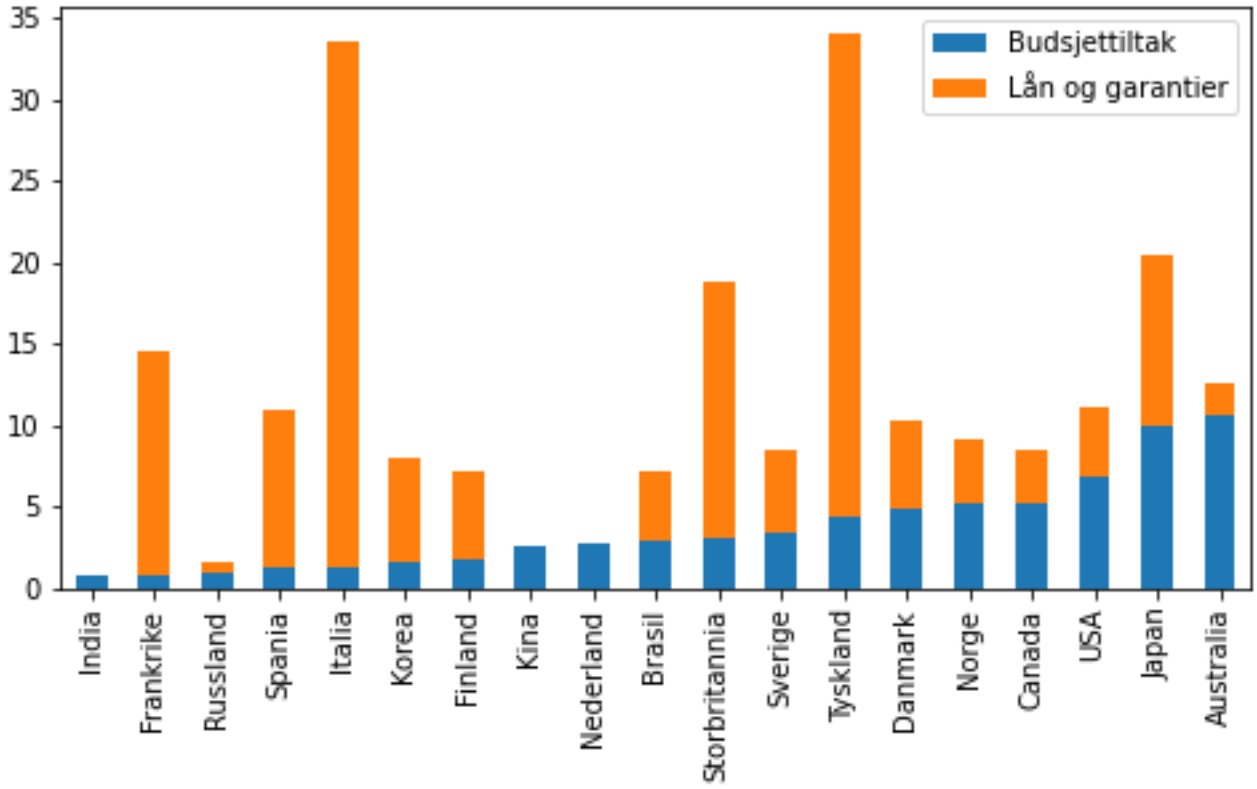

Figure 1: Figuren som tegnes av programmet ovenfor

- `# Les filen kap1.xlsx`: Alle linjer som starter med `#` er kommentarer. Disse blir ignorert av Python. Du kan bruke disse for å gjøre det enklere for deg selv å få oversikt over programmet ditt.
- `pd.read_excel()`: Leser inn Excelfiler. Merk at en kommando avgrenses av parantesene, slik at linje 4 til 11 er en enkeltkommando med flere parametre. Du angir først filnavnet (`kap1.xlsx`). Deretter kan du angi opsjoner som beskriver hvordan filen skal lastes inn. Her sier du at du vil laste inn arket med navnet `1.2` og beskriver hvordan de forskjellige radene og kolonnene skal tolkes. En litt forvirrende ting er at Python teller fra 0, så for eksempel `header=4` betyr at rad 5 i Excelarket skal brukes som overskrift.
- `print(budsjett)`: Skriver tallene til skjermen. Legg merke til at `budsjett` her refererer til hele arket og at dataene skrives ut som en oppstilt tabell
- `budsjett.plot.bar()`: Tegner et barplott av dataene.

Når du kjører programmet ditt vil du se tallene nede til høyre. For å se plottet må du klikke på **Plots** i vinduet oppe til høyre i Spyder.

Som tidligere, bruk litt tid på å leke med kommandoene. Prøv å endre på de forskjellige opsjonene til `pd.read_excel()` og se hvordan tabellen endrer seg. Hva skjer om du

bruker `plot.barh()` eller `stacked=False` i `budsjett.plot.bar()`-linjen?

pandas er et veldig kraftig verktøy, og du vil få se mange av mulighetene i selve kurset.

Kart

Mange data har en naturlig posisjon slik at det vil være nyttig å vise dem på kart. Python gjør det mulig å jobbe med forskjellige typer kart:

- Statiske kart hvor du designer et fast utsnitt
- Dynamiske kart hvor brukeren kan zoome og flytte rundt på kartet, samt klikke på datapunkter for å utforske dem nærmere.

I kurset vil du se hvordan du kan lage egne **dynamiske kart**. Til dette skal du bruke pakken Folium. Denne pakken er ikke en del av standardinstallasjonen til Anaconda. Folium må installeres som en ekstra pakke.

I **Anaconda Navigator** kan du gjøre følgende:

- Klikk **Environments** i listen til venstre
- Klikk **Channels**
- Klikk **Add...** og legg til URL'en `https://conda.anaconda.org/conda-forge/`
- Trykk **Enter** og klikk **Update channels**
- Velg **All** i nedtrekksmenyen og søk etter **folium**
- Marker **folium**
- Klikk **Apply**

(Ana)conda kan ha problemer med brannmur og lignende. Om installasjonen ser ut til å stoppe helt kan det være et problem med at Anaconda ikke kommer seg på nett. Det kan hjelpe å koble seg til et mer åpent nett hvis mulig.

Du kan sjekke at installasjonen av Folium har gått bra, ved å starte en ny fil i Spyder. Kall denne `kart.py` og skriv inn følgende kode:

Python

```python
import folium

kart = folium.Map()
kart.save("kart.html")
```

Kjør koden. Dersom koden ikke gir noen feilmelding har du installert Folium og er klar til kurset.

Legg merke til at dette lille programmet lager en fil som heter `kart.html`. Du kan dobbelklikke denne filen i en filutforsker for å åpne den i nettleseren din. Du vil se et verdenskart som du kan navigere rundt i.

Dersom du **ikke** fikk til å installere Folium gjennom Anaconda Navigator kan du bruke et kommandovindu i stedet:

- Åpne **Anaconda Prompt** fra startmenyen
- Skriv `conda install folium -c conda-forge`
- Om `conda`-kommandoen gir feilmelding, kan du prøve `python -m pip install folium` i stedet.