## Forberedelser til workshop for Python og KI.

**Steg 1 – 5 er en éngangsoperasjon vi må gjennom for å sette opp et egnet analysemiljø.**

**Har utarbeidet en detaljert veiledning, men har forståelse for at det fortsatt kan være utfordrende for noen. Gjør ditt beste, og send meg gjerne en epost (**[**even.sannes.riiser@telemarkfylke.no**](mailto:even.sannes.riiser@telemarkfylke.no)**) hvis du står fast! 😊**

Forberedelsene innebærer følgende steg:

1. Installere nødvendig programvare
2. Sette opp editoren hvor vi skal skrive kode (med god hjelp av KI)
3. Koble oss til Github (slik at vi kan jobbe i «skyen»)
4. Forberede et «virtuelt miljø» i Python med nødvendige programvare.
5. Opprette ditt eget virtuelle miljø hvor du skal behandle og analysere data.
6. **Installasjon av programvare**

* Windsurf Editor (<https://codeium.com/windsurf>)
  + **Windsurf**: Har en integrert AI-assistent (Cascade) som kjenner og kan redigere alt du har av kode, scripts osv. Anbefales, men også mulig å bruke:
  + **Visual Studio Code** (VSC): Basert på VSC, og bruker stort sett samme utvidelser. Har Github copilot, men ikke en like godt integrert AI-assistent.

*AI-funksjonene (Cascade/Github copilot) har et begrenset antall gratis interaksjoner. Anbefaler å teste en måned med Windsurf (15 USD), men ikke et must.*

* Anaconda (= Python) (<https://www.anaconda.com/download>) (må oppgi epost)
* “Git for Windows” (<https://gitforwindows.org>)

1. **Sette opp kode-editoren Windsurf**
2. Installere følgende utvidelser/extensions (ikonet ):

* Python
* Jupyter
* indent-rainbow
* Rainbow CSV
* Material Icon Theme. Klikk “Set file icon theme”: Et bilde som inneholder tekst, skjermbilde, Font

  Automatisk generert beskrivelse
* Hvis VSC: Github copilot (Logg så inn på Extension.)

1. I Windsurf eller VSC: File --> Preferences --> Settings**: Sørg for at denne er huket av (enabled)!** (Tips: Søk deg fram til innstillingen i søkefeltet.)



1. **Sette opp kobling mot Github**

*GitHub sørger for sikker lagring av prosjektene dine i skyen, samtidig som det gjør csv-filer ol. tilgjengelig via en url. Dette kan være nyttig i mange sammenhenger. GitHub gjør det også enkelt å samarbeide på kode ved å spore endringer, og alle i teamet får tilgang til det samme området (repositoriet).*

1. *Hvis du ikke allerede har det*, opprett et repository på din Github-konto
   1. Logg inn på din Github (i nettleseren)
   2. Repositories --> New

Et “repository” er et **område på Github** som er tilknyttet et prosjekt. Her havner blant annet filer og mapper tilknyttet prosjektet.

1. I Windsurf, velg fane “Source control” () i verktøylinjen langs venstre side av skjermen, “Clone repository” og “Clone from Github”. Mulig du så må logge inn på egen Github-konto. Da kan du velge blant egne repositories.

Et bilde som inneholder tekst, skjermbilde, Font

KI-generert innhold kan være feil.

Et bilde som inneholder tekst, skjermbilde, Font, programvare

KI-generert innhold kan være feil.

Alternativt kan du bruke url (..git) som du finner i repositoriet i nettleseren.

Et bilde som inneholder tekst, skjermbilde, Font, programvare

KI-generert innhold kan være feil.

1. En lokal kopi av repositoriet ditt åpnes nå i mappesystemet i Windsurf. Noter deg plasseringen (banen) til din lokale kopi av Github-repositoriet. **Tips:** Velg mappen i Windsurf, høyreklikk og velg “Reveal in File Explorer”.

Eks: Mitt repository på Github heter “Telemark”, og mappen på min lokale PC er **(…)/OneDrive - Telemark fylkeskommune/Github/Telemark**

1. Sett opp følgende mappestruktur i ditt repository (filer i **rødt**, mapper i **svart**). Filer og mapper er **vedlagt denne veiledningen** som en zip-fil.

For ordens skyld: I rota ligger det altså én fil (.gitignore) og én mappe (Python).

Et bilde som inneholder tekst, Font, diagram, skjermbilde

KI-generert innhold kan være feil.

**Kjekt å vite:** Når github er satt opp vil alle endringer du gjør i kode/filer/mapper registreres, og kan synkes til repositoriet under fane “Source control” () ved å legge inn en “Message” og trykke “Commit” --> “Sync”. Eksempel:

Et bilde som inneholder tekst, skjermbilde, programvare, Multimedieprogramvare

KI-generert innhold kan være feil.

1. Filen .gitignore definerer innhold vi ikke ønsker å laste opp på Github, f. eks. filer med sensitivt innhold som nøkkelkoder, passord osv. Sjekk derfor at filen inneholder:

Et bilde som inneholder tekst, Font, skjermbilde, nummer

KI-generert innhold kan være feil.

1. Nå er et passende tidspunkt å commite/laste opp endringer til Github. Gå til “Source control” (), skriv en valgfri kommentar, og trykk “Commit” --> “Sync”. Sjekk gjerne om de nyeste endringene er oppdatert på nettsidene for ditt repository.
2. For at et script skal kunne jobbe med filer som allerede ligger på Github (eks. “data.csv”) må vi opprette en “token” (nøkkelkode). Dette fordi brukernavn og passord ikke egner seg i kode/scripts.
   1. I Github, velg “Settings (for *konto*, ikke repository) --> Developer settings --> Personal access tokens --> Tokens (classic) --> Generate new token --> Generate new token (classic)”.
   2. Sett ønsket varighet (f. eks. evig), huk av for “repo” og copy/paste din token inn i en fil du kaller “**token.env**”.

GITHUB\_TOKEN="din\_token\_mellom\_anførselstegnene"

* 1. Legg “token.env” i mappen “Python”.

**PS:** Nøkkelen i “token.env” er sensitiv informasjon. Ved å legge en peker til “token.env” i fila “.gitignore”, sørger vi for at Github “ignorerer” nøkkelen når den overvåker filer og endringer. “token.env” blir derfor kun liggende på din lokale PC.

1. **Forberede et virtuelt miljø (conda environment) hvor vi vil jobbe.**

*Environments i conda er frittstående, isolerte miljøer hvor man kan installere en spesifikk versjon av Python og diverse software-pakker. Programmer (= pakker eller libraries) i Python (eks. pandas) er ofte avhengig av andre programmer, og miljøet sørger for at avhengighetene går knirkefritt, også ved oppdateringer av enkeltprogrammer. I et environment kan man installere akkurat de programmene og verktøyene man trenger. For å sette opp et conda environment må vi:*

1. Sørge for at conda er lagt til i Windows miljøvariabel “PATH”. Her forteller vi Windows hvor den skal lete etter vår “Python interpreter”. Startmeny --> Skriv “miljøvariabler” --> Velg “Rediger miljøvariabler for kontoen din”. Under brukervariabler for din bruker, merk “Path” og velg “Rediger”. Trykk “Ny”, og lim inn bane til mappe “anaconda3”. Finnes som regel enten under

“**C:\ProgramData\anaconda3**” eller “**C:\Users\brukernavn\AppData\Local\anaconda3**”. Trykk OK x2.

1. Forberede installasjon av environment vha. fila “environment.yaml”. Denne inneholder:
   1. Alle pakker vi ønsker å installere (flere kan installeres senere).
   2. Noen viktige variabler i Python-miljøet vårt. Disse peker til mapper vi bruker ofte. Slik slipper vi å “hardcode” full filbane i scriptene hver gang vi ønsker å peke til mye brukte filer og mapper. Vi er heller ikke så sårbare for tilfeller hvor vi har behov for å flytte scripts, fordi vi slipper å endre på banene.

Åpne “environment.yaml” og endre de tre siste linjene til å matche mappestrukturen på din lokale PC (**se pkt. 3C**). **Merk doble backslasher!:** \\

**Tips: Studer innholdet i fila nøye.**

**(NB:** Hvis dere er flere som deler samme repository: Alle lager en mappe med sitt navn i mappen “Python”. Her legger man så sin “token.env” med personlig nøkkelkode, samt sin “environment.yaml” med siste 3 linjer tilpasset egen mappestruktur. Pek til korrekte baner i pkt. 3E og 5D. Ta kontakt ved spørsmål!)

1. **Sette opp et “conda environment” i Windsurf.**
2. Åpne kommandopalett i Windsurf ved å trykke “Ctrl” + “Shift” + “P”, skriv/velg “Python: Select interpreter”, vent evnt. noen sekunder, og velg “Python X.XX.X (‘base’)”.
3. Åpne en ny terminal (Terminal --> New terminal). NB: Sørg for at det er en vanlig “Command prompt” terminal, kan evnt. velge å åpne ny nederst til høyre.

Et bilde som inneholder tekst, skjermbilde, programvare

Automatisk generert beskrivelse

1. Du skal nå stå i mappen [ditt repository]. Åpne mappen “Python” ved å skrive “cd Python”. (Viktige kommandoer: **dir**: vis innhold, **cd mappenavn**: gå inn i mappe, **cd..** : gå tilbake)
2. Nå skal vi opprette analysemiljøet ("analyse") vi skal jobbe i, hvor alle programmer (pakker) installeres. All info vi trenger ligger i "environment.yaml". Opprettelsen kan ta noen minutter.

Skriv følgende i terminalen: **conda env create --file environment.yaml**

Trykk “Y” ved forespørsel. Vent til prosessen er fullført.

1. Gå inn i miljøet ved å skrive **conda activate analyse** i terminalen. Ønsker man å se installerte pakker, skriver man **conda list**.
2. Åpne kommandopalett i Windsurf ved å trykke “Ctrl” + “Shift” + “P”, skriv/velg “Python: Select interpreter”, vent evnt. noen sekunder, og velg “Python X.XX.X (‘**analyse’**)”.

# Endelig tid for å leke seg i Python! :)

**Øvelse 1: Interaktivt vindu og in-line forslag til kode**

Husk: Vær i environment “analyse” (conda activate analyse)

1. Legg vedlagte script “**ssb\_test.py**” i mappen **Scripts**. Studer scriptet nøye.
2. Marker all kode overfor linja “Leke seg med datasett”, og trykk **Shift + Enter** for å kjøre koden. Vent til resultatene dukker opp i eget “interaktivt vindu”. Her kan man altså kjøre kode bit for bit, og få output direkte i eget vindu!
3. Dobbeltklikk variabelen “**df**” i koden din og trykk Shift + Enter. Da vises en “penere formatert” versjon av datasettet enn hvis du kun skriver “**print(df)**”. Gull verdt når man jobber med uoversiktlige datasett!
4. Sett musepekeren på kodelinje 83 (“**df.info()**”) og trykk Shift + Enter. Da kjøres kun denne linjen.
5. Sett musepeker på tom linje 85, og skriv “**## Vis unike verdier i kolonne “alder”**”. AI-funksjonen kommer opp med forslag fortløpende. Den prøver å fullføre kommentar, og foreslår etter hvert også kode. Trykk “Tab” for å autofullføre/godta forslag.
6. Kjør foreslått kode ved å sette markør på linja og trykke Shift + Enter (eller marker koden og trykk Shift + Enter).
7. Kjøre resten av koden og sjekk at du får ut en .csv-fil.

**Merk:** Github-funksjonaliteten i Windsurf registrerer alle oppdateringer i kode og nye filer som genereres. Neste gang man “committer” (pkt. 3.5) opplastes/oppdateres alt på Github.

**Øvelse 2: “Cascade del 1 – Redigere eksisterende kode”**

[*Cascade*](https://codeium.com/cascade) *er en "AI-agent” som hjelper deg å kode. En slags “ChatGPT” innbakt i selve kode-editoren. Den kjenner hele “kodebasen” din (dvs. alle filer og scripts du har i mappesystemet/repositoryet ditt), og kan foreslå ny kode eller endringer i eksisterende kode basert på det du ønsker å få til. Dette innebærer også “multi-file editing”, altså endringer i flere filer av gangen.*

*Cascade tilbyr to funksjoner; “Chat”, hvor man kan diskutere og få forslag, eller “Write”, som kan gå inn og endre koden din direkte (hvis du godtar forslagene). Jeg bruker mest “Write”. Cascade kan også kjøre kode i terminalen, men dette fungerer ikke alltid like godt når man er i et conda environment. Jeg markerer heller teksten og kjører vha. Shift + Enter.*

*Cascade husker tidligere samtaler, støtter innliming av screenshots, tilbyr flere ulike modeller og mye annet. Den forstår både norsk og engelsk, men hvis du er komfortabel med engelsk, ville jeg valgt det.*

1. Velg “Write”-funksjon i Cascade, og skriv:

In the script “ssb\_test.py”, I want to fetch the data for the two most recent years (not only the latest). Also, remove all rows from the dataframe “df” where “kjønn” equals “Begge kjønn”.

1. Vent til Cascade har jobbet ferdig, og se over forslag til endringer. Ny kode i grønt, kode som vil fjernes i rødt. For eksempel:



1. Hvis du er fornøyd, velg “Accept” (enkelt-endring) eller “Keep changes” (alt), og kjør koden på nytt (vha. Shift + Enter).
2. Inspiser resultatet. Ser det riktig ut?
3. Hos meg foreslo den “Kjønn” istedenfor det korrekte “kjonn”. Hvis du får en feilmelding kan du kopiere denne (og eventuelle andre outputs) ved å trykke på de tre prikkene (…) i det interaktive vinduer, velge “Copy cell output”, og lime inn i Cascade. Gjerne med en kort beskrivelse. Ofte kommer man nærmere en løsning – i hvertfall i mitt tilfelle (se under). 😊

Et bilde som inneholder tekst, skjermbilde, Font

KI-generert innhold kan være feil.

gir:

Et bilde som inneholder tekst, Multimedieprogramvare, Font, programvare

KI-generert innhold kan være feil.

1. Kjør scriptet en siste gang, inspiser resultat, og last opp til Github.

**Øvelse 3: “Cascade del 2 – Opprette kode fra bunnen av”**

1. Velg “Write”-funksjon i Cascade, og skriv:

“Make a script to translate text to Yoda speech based on the API described on this page: <https://funtranslations.com/api/yoda>”

1. Velg å godta den foreslåtte koden, før du kjører den.
2. Skriv inn valgfri tekst på engelsk, og sjekk resultatet. “So much fun, coding with Windsurf is!”

**Øvelse 4: Hente data fra Github**

I denne øvelsen skal vi forsøke å bruke csv-filen vi har lastet opp på Github (i oppgave 2) som input i et nytt Python-script.

Alle filer på Github har en url, og det kan for eksempel være nyttig å peke til slike (spesielt csv-filer) i andre scripts for videre dataanalyse, eller som datakilde i figurer i blant annet Power BI.

Velg «Chat»-mode nede til høyre på skjermen i Cascade (dvs. AI-assistenten). Skriv for eksempel følgende (du kan også skrive på norsk):

1. I want to use a csv file in my Github repository as input in a Python script. What do I need to do?

Du vil sannsynligvis få beskjed om at du vil trenge en «token». Da ville jeg opplyst om følgende:

1. I have my Github token in a separate file called "token.env". How can I use this in the script?

Etter hvert som du får forslag til kode, kan du opprette en tom Pythonfil (.py), og velge «Insert». Da dukker forslaget opp i filen din, og du kan samarbeide videre om koden med AI-assistenten i «Write» mode.

Spør og grav - først AI’en, så Even! 😊

Klarer du å lese inn innholdet fra «ssb\_data.csv» på Github til en pandas data frame? 😉