En praktisk guide for nybegynnere

BYGG DIN EGEN KI-AGENT



KENNETH BAREKSTEN

Innhold

HVA ER EN KI-AGENT?

Side 4

BYGG EN KI-AGENT?

Side 7

GI AGENTEN EN STIL

Side 10

HA EN INTERAKTIV SAMTALE

Side 12

LEGG TIL VERKTØY

Side 14

BRUK DOKUMENTER ELLER TEKSTFILER

Side 16

SKAP EN NYTTEAGENT

Side 18

TA ET SKRITT VIDERE

Side 20

GI AGENTEN EN FUNKSJON

Side 22

VANLIG FEILMELDINGER

Side 24

TA ANSVAR

Side 25

NESTE STEG VIDERE

Side 26

NØKKELORD OG KILDER

Side 27

Innledning

Det har aldri vært enklere å lage smarte systemer uten å være utvikler. Med språkmodeller som ChatGPT og verktøy som Googles Colab kan du bygge din egen agent: en liten digital hjelper som forstår meldinger, husker samtaler og bruker verktøy for å løse oppgaver.

Denne boka er skrevet for deg som vil lære litt om KI-agenter som et praktisk verktøy. Du bruker Google Colab og litt enkel bruk av programmeringsspråket Python, men du trenger ikke kunne kode fra før. Steg for steg med enkel kode og eksempler du kan teste underveis, bygger du den selv.

Du lærer hva en agent er, hvordan du styrer tonen og rollen, hvordan du bygger samtaler og hvordan du gir agenten tilgang til verktøy og dokumenter. Til slutt kombinerer du alt og lager en agent som kan brukes i til enkle oppgaver. Denne boka er skrevet med KI som assistent og lisensiert under Creative Commons. Det betyr at du kan dele, bruke og tilpasse innholdet, også kommersielt, så lenge du krediterer forfatteren. Siden jeg er lærer er mange av eksemplene skolerelaterte, men du står helt fritt til å eksperimentere med dette selv.

Denne boka passer for deg som:

✅ vil forstå hvordan KI-agenten fungerer

✅ vil bygge en agent du kan bruke igjen og igjen

✅ ønsker en praktisk innføring med ekte eksempler

✅ er nysgjerrig på hvordan Colab og Python brukes nyttig

Denne boka er ikke:

❌ en guide til avansert programmering

❌ et oppslagsverk for utviklere

❌ et ferdig verktøy du bare klikker på

Kenneth Bareksten, juni 2025





HVA ER EN KI-AGENT?

Det som skiller en agent fra en bot

Forestill deg at du har en digital assistent som gjør hele jobben. Ikke en passiv chatbot som spytter ut svar, men en kollega som tar oppgaven og leverer resultat. Du skriver: «Svar foreldrene til 9B, lag en kort oppsummering av siste matteprøve og legg ved en tydelig fremdriftsplan.» Minutter senere er e-posten klar til å sendes etter du har lest igjennom. Du har ikke rørt musa ved siden av deg. Det er en KI-agent.

Det er likevel viktig å skille de ulike systemene fra hverandre. En chatbot svarer bare på spørsmål. En KI-agent gjør en konkret oppgave for deg. Og en agentisk KI kan tenke mer selvstendig og løse komplekse problemer på egen hånd. Her er en enkel oversikt:

Type	Kort forklaring	Eksempel
Chatbot	Svarer på spørsmål. Fungerer som en samtalepartner uten egne handlinger.	Chatbot som svarer foresatte på spørsmål om timeplan og fravær.
KI-agent	Utfører en konkret oppgave på dine vegne. Forstår mål, bruker verktøy, tar valg.	Agent som henter relevante læringsressurser og lager forslag til undervisningsopplegg.
Agentisk KI	Planlegger, deler opp oppgaver og justerer underveis. Handler som en digital kollega.	KI som selv lager en hel undervisningsplan med oppgaver og forslag til justeringer underveis.

En agent, i motsetning til en vanlig chatbot, analyserer språket ditt og skjønner hva du egentlig vil. Du slipper å bruke spesifikke kommandoer eller formulere deg på en bestemt måte. Agenten kan koble seg til verktøy, systemer eller API-er og utføre oppgaver for deg – som å sende e-post, hente ut data eller fylle ut skjemaer. Den jobber på egen hånd, vet når den er ferdig, oppdager feil underveis og tilpasser seg uten at du må styre alt.

For at en KI-agent skal fungere, må tre ting være på plass. Først trenger den en **hjerne**. Hjernen i denne sammenhengen er en språkmodell, som for eksempel GPT-4. Det er denne som forstår hva du skriver, tolker hensikten din og resonerer seg fram til neste steg.

Deretter trenger agenten **instruksjoner**. Dette er oppskriften som forteller hva den skal gjøre og hvordan den skal oppføre seg. Det kan være alt fra "svar høflig" til "be brukeren om ordrenummer hvis det mangler".

Til slutt trenger agenten **verktøy**. Den må kunne gjøre mer enn å bare snakke, som å sende e-post, hente værdata eller oppdatere et regneark. Verktøyene gir agenten handlekraft. Når du setter hjerne, instruksjoner og verktøy tilsammen får du en fungerende KI-agent.

Når bør du bruke en agent?

Det er ikke alltid du trenger en KI-agent. Noen oppgaver er enkle nok til at vanlige skjemaer, apper eller faste rutiner fungerer helt fint. Har du et fast sett med regler og lite variasjon rundt disse, holder ofte de vanlige systemene du allerede bruker. Det går raskt og det er trygt så lenge

situasjonen er forutsigbar. Mange oppgaver i arbeidshverdagen er likevel det motsatte. Informasjonen kan være uklar, ustrukturert eller fulle av unntak. De krever vurderinger som ikke alltid passer inn i faste skjemaer. Her sliter tradisjonelle verktøy. Da kan en agent være en smartere løsning.

Her er tre situasjoner der en agent kan være aktuelt:

1. Når avgjørelsen krever vurdering eller skjønn
En lærer skal lage forslag til aktiviteter i kroppsøving for en uke med dårlig vær. Agenten kan ta hensyn til temaet, tilgjengelige lokaler og tidligere forslag, og komme med konkrete idéer

2. Når reglene er blitt uoversiktlige
Eksempel: Et HR-system med hundrevis av interne rutiner og unntak. En agent kan navigere dette lettere enn et skjema med avkrysningsbokser.

3. Når dataene er ustrukturerte
Eksempel: En kommuneansatt får e-poster med vedlegg, lange forklaringer og fritekst. En agent kan trekke ut datoer, kategorisere saken og foreslå tiltak.

Eksempel på agentvirksomhet

En kunde skriver: "Hei, hvor er pakken min?"
Agenten:

- Leser meldingen
- Sjekker ordrestatus
- Ser at pakken er på vei
- Svarer: "Pakken din kommer fredag. Her er sporingsnummeret."

GDPR og personvern

KI-agenter kan virke både imponerende og litt skumle. Mange lurer på om det egentlig er greit å bruke dette i skolen. Svaret er ja, men med ansvar. Når du bruker en KI-agent, er det du og ikke maskinen som har ansvaret for hva den gjør. Det gjelder særlig hvis agenten jobber med sensitive person-opplysninger. Du må være like bevisst som når du bruker en e-post eller lærerplattform.

Noen enkle tommelfingerregler å huske på:

- Bruk aldri elevnavn eller sensitive opplysninger direkte i agenten.
- Dobbeltsjekk alltid det agenten produserer før du sender det videre.
- Ikke la agenten ta avgjørelser som krever faglig eller etisk skjønn.

Det finnes regler, som Personopplysningsloven, som beskytter personvernet. Disse gjelder uansett om du bruker en agent eller ikke. Poenget er at agenten er et verktøy og du må bruke det som en profesjonell.

I Norge gjelder personopplysningsloven, som bygger på EUs regelverk for personvern kjent som GDPR. Dette regelverket handler om å beskytte personopplysninger, altså informasjon som kan knyttes til enkeltpersoner, for eksempel navn, karakterer eller spesielle behov. Når du bruker en KI-

agent, er det viktig å huske at du ikke skal skrive inn elevdata eller annen sensitiv informasjon, med mindre du har et klart behandlingsgrunnlag. Det betyr at du gjerne kan bruke agenten til å skrive generelle utkast, forbedre tekster eller generere forslag, men ikke til å håndtere informasjon som angår navngitte elever.

Et godt prinsipp er å spørre deg selv: Tåler dette å stå på en plakat i gangen utenfor klasserommet? Hvis ikke, bør det heller ikke inn i agenten.

Hvordan ser en agent ut i praksis?

Tenk deg at du skal forberede et foreldremøte og vil lage en foreløpig agenda. Du åpner Colab og skriver til agenten:

Lag forslag til en enkel agenda for foreldremøte på 8. trinn.
Det skal vare én time og inneholde noe om fag, trivsel og fravær.

Agenten svarer:

1. Velkommen og introduksjon (5 min)
2. Kort om faglig utvikling og vurdering (15 min)
3. Trivsel i klassen og sosiale tiltak (15 min)
4. Fravær og samarbeid hjem-skole (10 min)
5. Spørsmål og diskusjon (15 min)

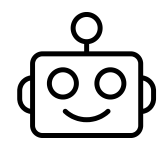
Her bruker agenten ingen eksterne data. Den kombinerer språkforståelse med innebygd kunnskap om struktur og skolekontekst – basert på beskrivelsen du ga. Du kan deretter redigere, bytte ut eller sende agendaen videre.

Neste kapittel

Nå som du vet når en agent er nyttig, skal vi i neste kapittel bygge vår første agent helt fra bunnen av. Du får prøve selv, steg for steg, og vi starter med noe alle kan få til.

TIPS! Før du går videre:

- Opprett en gratis bruker på Google Colab. Det er her du skal teste og kjøre koden i resten av boka. Kodingen foregår direkte i nettleseren. Du trenger ikke å installere noe.





BYGG EN KI-AGENT

I denne delen går vi fra teori til praksis. Du skal lage din egen KI-agent, og som du kan teste selv i nettleseren. Det høres kanskje teknisk ut, men vi guider deg steg for steg, uten at du trenger å kunne kode fra før.

Du får lære hvordan du skriver enkel kode i Google Colab, hvordan du bruker OpenAI sin modell, og hvordan du får agenten til å gjøre noe nyttig, som å svare på forespørsler, oppføre seg høflig og huske konteksten.

Det du trenger til dette kapitlet

- En Google-konto (for å bruke Colab)
- En API-nøkkel fra OpenAI

Steg 1: Åpne Google Colab

Gå til https://colab.research.google.com. Logg inn hvis du ikke allerede er det. Klikk på File > New notebook for å lage en ny Colab-fil.
Gi den gjerne navnet: Min første KI-agent

Steg 2: Skriv første kodelinje

I den første cellen limer du inn dette:

```
!pip install --upgrade openai
```

Trykk Shift + Enter for å kjøre koden.
Dette installerer pakken vi trenger for å bruke OpenAI sin språkmodell.

Steg 3: Få en API-nøkkel fra OpenAI

For å kunne bruke modellen, må vi bevise for systemet at vi har tilgang. Det gjøres med en såkalt API-nøkkel, en slags passordnøkkel du limer inn i koden. Du kan bruke andre språkmodeller, men jeg har valgt OpenAI.

Hva er en API-nøkkel?

En API-nøkkel er en hemmelig kode som gir deg tilgang til OpenAI sine tjenester (du kan bruke en språkmodell). Du får din egen personlige nøkkel, og den brukes hver gang agenten sender en forespørsel.

Slik får du en nøkkel:

1. Gå til https://platform.openai.com
2. Trykk på Sign up hvis du ikke har konto, eller
3. Logg inn hvis du allerede har en.
4. Gå til menyen oppe til høyre og klikk på Dashboard
5. På venstre side i menyen trykker du på API keys
6. Trykk på + Create new secret key
7. Kopier nøkkelen og ta vare på den! Du ser den kun én gang.

Steg 4: Sett opp tilgang i Colabs hemmelig oppbevaring

1. Trykk på nøkkelsymbolet på venstre side i Colab
2. På navn så skriver du: OPENAI_API_KEY
3. På verdi så limer du inn API-koden fra OpenAI
4. Kryss ut fanen etterpå
5. Start kodingen med koden under:

```
import openai
from google.colab import userdata

openai.api_key = userdata.get("OPENAI_API_KEY")
```



TIPS: Viktig å huske om API og hemmelig oppbevaring

- Legg aldri API-nøkkel rett i koden. Den kan misbrukes.
- Bruk Colabs hemmelige oppbevaring, følger prosjektet ditt så lenge du er logget inn med samme Google-konto og bruker samme fil.

Steg 5: Send første forespørsel

Nå skal vi sende vår første melding til språkmodellen, altså be agenten gjøre noe. Det gjør vi ved å definere en liste med meldinger i koden. Du sender en melding fra brukeren (deg selv) ved å bruke rollen "user" og skrive meldingen som tekst. Modellen tolker meldingen og svarer tilbake med "assistant"-rollen. Du kan også bruke "system"-rollen til å sette tonen eller instruksjoner for hvordan agenten skal oppføre seg.

En forespørsel består av to deler:

- En rolle: Hvem er det som snakker – "system", "user" eller "assistant"
- En melding: Hva de sier

I eksempelet under gir vi først modellen en rolle (systembeskrivelse), og så sender vi en melding som bruker (user):

```
chat_completion = openai.chat.completions.create(
    model="gpt-4o-mini",
    messages=[
        {
            "role": "system",
            "content": """Du er en høflig assistent som hjelper lærere."""
        },
        {
            "role": "user",
            "content": """Kan du skrive et kort og positivt svar til foreldrene til 9B om
matteprøven?"""
        }
    ]
)

print(chat_completion.choices[0].message.content)
```

Dette skjedde nå

Du har laget en enkel agent som tolker instruksjoner og svarer som en hjelpsom kollega. Bak svaret ligger mye: modellen forstår hensikten din, henter relevant kunnskap og formulerer et passende svar.

Agenten din kan i denne formen brukes til enkle skriveoppgaver og meldingshåndtering. Du kan teste ulike formuleringer, som:

- "Skriv en positiv og motiverende kommentar til en elev som har forbedret seg i matematikk."
- "Formuler en høflig påminnelse om innleveringsfrist for uke 24."

Kan jeg spørre om mer?

Ja, men ikke som du kanskje er vant med. I motsetning til ChatGPT og andre KI-verktøy, finnes det ikke noe chattefelt i Google Colab. Du skriver spørsmålet ditt direkte inn i Python-koden. Det kan føles litt teknisk i starten, men du blir fort vant til det. Alt du vil spørre agenten om må nemlig skrives direkte i meldingslisten (messages).

Det betyr at du må endre teksten i "content"-delen, kjøre cellen og se svaret skrevet ut nedenfor hver gang. Hvis du vil stille et nytt spørsmål, endrer du bare teksten i "content" og kjører koden på nytt.

Feil du kan møte

Selv om oppsettet er enkelt, kan det dukke opp småfeil. Her er noen vanlige problemer og hvordan du løser dem. En mer omfattende feilliste bakerst:

- **API-nøkkelen virker ikke**: Dobbeltsjekk at du har limt inn nøkkelen riktig under “Secrets” i Colab. Navnet må være OPENAI_API_KEY.
- **Rate limit eller stopp i Colab**: Prøv å vente noen sekunder og kjøre cellen på nytt. Hvis det skjer ofte, bytt til en billigere modell (gpt-3.5-turbo) under utvikling.
- **Pakke mangler**: Hvis openai eller docx ikke finnes, kjør !pip install openai eller !pip install python-docx på nytt.

Kostnader

Hver gang du bruker agenten, sendes en melding til OpenAI og det koster litt. GPT-4o er dyrere, mens gpt-3.5-turbo er et rimeligere alternativ for testing. Et vanlig svar koster bare noen øre, men det kan bli mye hvis du tester ofte. Du får oversikt på platform.openai.com.

Neste kapittel

I neste kapittel skal du lære hvordan du gir agenten personlighet og stil. Du får teste hvordan én setning i starten kan gjøre stor forskjell: Skal agenten være høflig, ungdommelig, morsom eller streng? Du lærer hvordan du skriver gode systembeskjeder og får agenten til å oppføre seg akkurat slik du vil.



TIPS! Før du går videre, dobbeltsjekk at du har:

- En fungerende Colab-notatbok
- Installert OpenAI-pakken uten feil
- En gyldig API-nøkkel fra OpenAI limt inn i koden
- Fått modellen til å svare deg på en enkel forespørsel



GI AGENTEN EN STIL

Vi gir den en ny personlighet og stil

Du har nå laget din første KI-agent. Den kan svare på enkle forespørsler, men det føles kanskje litt generelt? Hva om du vil at agenten skal være ekstra vennlig eller ufin. Hva med at den skriver som en rådgiver, en lærer, eller en robot med sans for humor? Da trenger du bare én ting: en god systembeskjed.

Hva er en systembeskjed?

En systembeskjed er den første beskjeden du gir agenten, og fungerer som et manus. Den bestemmer hvordan agenten skal oppføre seg – hvilken rolle den har, tonen den bruker og hvordan den svarer. Du kan for eksempel skrive: “Du er en hjelpsom assistent som hjelper lærere,” eller “Du er en ungdommelig rådgiver som bruker emojis.” Dette påvirker om agenten svarer formelt eller uformelt, kort eller langt, og hvilket språk den bruker. Små justeringer kan gi store utslag i svarene.

Slike beskrivelser påvirker om svarene blir formelle eller vennlige, korte eller utdypende, og om agenten bruker teknisk språk, barnespråk, emojis eller fagspråk. Små endringer i systembeskjeden kan føre til helt ulike svar fra modellen. Vi skal nå prøve ut de tre eksemplene over.



TIPS! Når du tester kodene under, har du to valg:

- kopiere hele **chat_completion**-koden og lime den inn i en ny celle i Colab (test 1-3 under)
- endre teksten i **content**-feltet, altså det som står i "**system**" eller "**user**"-meldingene

Begge deler fungerer like bra. Hvis du vil beholde flere versjoner og sammenligne dem, er det lurt å kopiere og lime inn nye celler. Hvis du bare vil teste én ting om gangen, kan du bruke samme celle og endre teksten underveis

Test #1: Den vennlige rådgiveren

```
chat_completion = openai.chat.completions.create(
    model="gpt-4o-mini",
    messages=[
        {
            "role": "system",
            "content": """Du er en vennlig rådgiver som hjelper elever med skolearbeid.
Svar tydelig og oppmuntrende."""
        },
        {
            "role": "user",
            "content": """Jeg klarer ikke å bli ferdig med innleveringen.
Hva bør jeg gjøre?"""
        }
    ]
)

print(chat_completion.choices[0].message.content)
```

I eksemplet over sier vi til agenten at den skal være en vennlig rådgiver. Det gjør vi med system-meldingen. Når du så spør om hjelp, svarer agenten i en hjelpsom og positiv tone fordi den har fått en tydelig rolle å spille

Test #2: En streng historielærer fra 1920

```
chat_completion = openai.chat.completions.create(
    model="gpt-4o-mini",
    messages=[
        {
            "role": "system",
            "content": """Du er en streng historielærer fra 1920.
Du bruker gammeldags språk og er opptatt av disiplin og korrekt grammatikk."""
        },
        {
            "role": "user",
            "content": """Fortell meg hvorfor første verdenskrig startet."""
        }
    ]
)

print(chat_completion.choices[0].message.content)
```

Nå gir vi agenten en helt annen personlighet. Med bare én setning endrer vi stilen fullstendig. Agenten bruker nå et mer gammeldags språk og en strengere tone, fordi den tror den er en historielærer fra 1920. Dette viser hvor mye systembeskjeden styrer hvordan agenten svarer.

Test #3: Ungdommelig stil med emojis

```
chat_completion = openai.chat.completions.create(
    model="gpt-4o-mini",
    messages=[
        {
            "role": "system",
            "content": """Du er en kul og positiv skoleassistent
som snakker ungdommelig og bruker emojis."""
        },
        {
            "role": "user",
            "content": """Kan du minne meg på å levere inn matteleksa?"""
        }
    ]
)

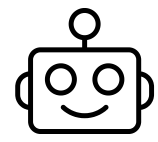
print(chat_completion.choices[0].message.content)
```

I dette eksempelet gir vi agenten en ungdommelig stil. Bare ved å endre én setning begynner den å bruke emojis, enklere språk og en mer avslappet tone. Dette viser hvor fleksibel agenten er og hvor lett du kan tilpasse den til ulike brukere.

Neste kapittel

Nå har du lært hvordan du kan gi agenten en rolle og få den til å svare slik du ønsker. Men hva skjer hvis du vil ha en lengre samtale? Hvordan kan agenten huske hva du skrev i forrige melding?

I neste kapittel skal vi bygge videre: Du lærer hvordan du lager en samtale med flere meldinger, hvordan agenten husker konteksten i en kort dialog, og hvordan du kan bruke dette til å simulere ekte samtaler.





INTERAKTIV SAMTALE

Slik husker agenten det du sier

Hittil har du laget en agent som svarer på én enkelt melding. Det er nyttig, men litt begrenset. Mange oppgaver krever mer enn ett spørsmål og ett svar. Du spør, får et svar, stiller et oppfølgingsspørsmål og får nye forslag.

I dette kapittelet lærer du hvordan du får agenten til å huske samtalen underveis. Du slipper å skrive alt selv. Python-koden tar vare på det du og agenten sier, og bygger opp samtalen automatisk. Det gjør at agenten kan svare mer sammenhengende akkurat som en ekte samtalepartner.

TIPS! Husk at agenten ikke har ordentlig hukommelse:

- Agenten husker ikke tidligere samtaler hvis du lukker Colab eller starter på nytt. Den starter alltid med blankt ark.
- Så lenge du holder samtalen åpen, husker agenten alt dere sier.

Eksempel:

Under ser du litt av en enkel versjon av agenten som du kan snakke med direkte i Colab. Du skriver inn én melding om gangen, og agenten svarer. Den husker hele samtalen så lenge du holder den i gang. Når du er ferdig, skriver du bare stopp.

```
# Start med en systembeskjed
messages = [
    {
        "role": "system",
        "content": """Du er en hjelpsom og positiv rådgiver som svarer kort og tydelig """
    }
]

# Sam
while
    u

    i


    m
```

⚠️ NB! Denne koden er litt mer avansert

Koden inneholder løkker, innrykk og lengre tekstblokker. Disse er avhengig av riktig innrykk og formatering. Hvis du kopierer direkte fra PDF-en, som i de enkle kodene over, kan det føre til feil.

Klikk derfor på lenken under over for å åpne en ren og uformatert tekstversjon du kan kopiere trygt inn i Colab.

KI-agent kodefiler/Interaktiv agent

Nå har du laget en agent som kan føre en ekte samtale og som husker det dere har snakket om, så lenge du holder koden i gang. Du trenger ikke skrive meldingslisten selv. Python tar vare på alt i bakgrunnen.

Dette gjør det mye enklere å teste ulike scenarier, stille oppfølgingsspørsmål og bygge en samtale som faktisk gir mening. Du kan bruke denne modellen som utgangspunkt for å utforske mer, uten å måtte kode på nytt hver gang.

Hvorfor svarte agenten slik?

Noen ganger vil du få svar som ikke treffer helt. Kanskje agenten misforstår spørsmålet, svarer i feil tone eller fokuserer på noe annet enn du hadde tenkt. Det betyr ikke at agenten er dum, det betyr bare at samtalen du sendte inn kanskje var uklar, mangelfull eller for bred.

Siden agenten bruker hele samtalen som grunnlag, kan én utydelig melding i starten farge hele svaret senere.



TIPS! Hvis noe ikke fungerer, prøv dette:

- Se gjennom hele samtalen. Hva har agenten fått med seg?
- Har du gitt tydelig rolle i systembeskjeden?
- Er meldingen din kort og konkret, eller vag og lang?
- Er du sikker på at du faktisk stilte et klart spørsmål?

En agent er god på mønstergjenkjenning, men den kan ikke lese tankene dine. Jo tydeligere du formulerer deg, jo bedre svar får du.

Hva hvis agenten misforstår?

Når du bygger en interaktiv samtale, kan det skje at agenten gir merkelige eller feilaktige svar. Kanskje den svarer på feil spørsmål, gjentar tidligere informasjon, eller finner på noe som ikke ble nevnt.

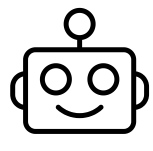
Dette skjer fordi agenten ikke «husker» på samme måte som et menneske. Den ser bare på teksten du har sendt inn i samtalen, og forsøker å gjette det mest sannsynlige svaret. Den leser meldinger som tekst, ikke som intensjoner. To viktige punkter:

1. Bruk en tydelig systemmelding. Fortell agenten hva den skal og ikke skal gjøre.
2. Skriv eksplisitte brukerbeskjeder. Når du følger opp, vis hva du mener.

Neste kapittel

Nå har du en fungerende samtaleagent som kan holde på en kontekst og svare sammenhengende. Det åpner mange muligheter, men agenten kan fortsatt bare snakke.

I neste kapittel skal du lære hvordan agenten kan gjøre noe mer enn å svare på tekst. Vi skal gi den et enkelt verktøy, slik at den kan hente inn informasjon fra andre kilder og bruke det i svaret sitt. Da begynner agenten virkelig å bli nyttig.





LEGG TIL VERKTØY

Få agenten til å gjøre noe mer

Hittil har agenten svart på det du skriver, basert på meldinger og samtalehistorikk. Det i seg selv er ganske kult, men det finnes grenser for hva den klarer alene. Den kan ikke hente ny informasjon, gjøre oppslag på nettet eller lese fra andre filer. Det du gir den i samtalen, er alt den vet. Men det kan vi gjøre noe med.

I dette kapittelet lærer du hvordan du kan koble agenten til et verktøy, noe som lar deg hente inn data utenfra og bruke det i svaret. Det er første steg mot en agent som ikke bare snakker, men faktisk gjør noe.

Hva mener vi med verktøy?

Når vi sier at vi skal legge til et verktøy, mener vi at agenten skal få tilgang til noe den ellers ikke har. Den kan skrive, forklare og svare, men den vet bare det du gir den. Den kan ikke hente ferske data fra nettet, åpne dokumenter eller gjøre utregninger på egen hånd.

Derfor trenger vi å hjelpe agenten litt. Vi bruker kode til å hente inn informasjon, som været akkurat nå, eller et tall fra en nettside og så lar vi agenten bruke det i svaret sitt. Det betyr at agenten får tilgang til oppdatert kunnskap, og kan svare bedre og mer konkret. Du kan tenke på det slik: Agenten snakker, men du gir den noe å snakke om.



TIPS! Fra svar til handling:

- Du kan bruke agenten til å koble sammen informasjon fra flere steder, som f.eks. timeplan, fravær og vær. Perfekt for å planlegge turer
- Verktøy gjør at agenten kan jobbe med sanntid, detaljer eller praktiske oppgaver som ikke bare forklarer, men også leverer.

Å legge til verktøy gjør at agenten kan gjøre mer enn å prate den kan faktisk bidra med noe konkret. Den kan trekke inn dagens valutakurs når du skal kjøpe utstyr til klassen, hente åpningstider fra et nettsted, eller lese en fil og oppsummere innholdet for deg.

Eksempel:

Nå skal du få teste hvordan du kan gi agenten oppdatert informasjon, i dette tilfellet værdata. Du skriver inn et sted, og agenten svarer med hjelp fra data som hentes direkte fra nettet. Dette er et enkelt, men godt eksempel på hvordan en agent kan bli mer enn bare en samtalepartner.

```
import requests
import openai
fro

# H
ope

pri

whi
    if sted.lower() == "stopp":
```

⚠️ NB! Last ned koden i TXT-format

Klikk på lenken under over for å åpne en ren og uformatert tekstversjon du kan kopiere trygt inn i Colab.

KI-agent kodefiler/Væragent

Her skjer det flere ting. Agenten forstår ikke været direkte, men den tolker teksten og foreslår passende aktiviteter. Når vi kobler sammen språkmodell og verktøy, kan agenten gjøre mer enn bare å svare, den kan handle på bakgrunn av data.

Neste kapittel

I neste kapittel skal vi se hvordan agenten kan bruke dokumenter og tekstfiler. Da kan du laste opp for eksempel en ukeplan, en artikkel eller et refleksjonsnotat og få agenten til å oppsummere, forklare eller foreslå videre arbeid basert på innholdet.



DOKUMENTER OG FILER

Få agenten til å lese tekst

Hittil har du sendt korte meldinger rett inn i agenten. Det fungerer fint når du stiller et spørsmål, gir et stikkord eller henter litt værdata. Men hva om du vil at agenten skal bruke noe lengre, som en fagtekst, en e-post, en ukeplan eller en tekstfil?

For det kan den. Du kan laste opp en fil i Colab, lese innholdet og sende det videre til agenten. Da oppfører agenten seg som om du har skrevet hele teksten selv og kan svare på den, oppsummere den eller hente ut viktige poeng.

Hva slags dokumenter kan du bruke?

I praksis kan agenten lese alle dokumenter som kan åpnes som tekst. Her er noen eksempler:

- **docx-filer:** Vanlige Word-dokumenter. Perfekt for fagtekster, ukeplaner eller e-poster du har lagret. Vi bruker dette formatet i eksemplene videre.
- **csv-filer:** Regneark lagret som komma-separert tekst. Brukes ofte til oversikter, lister, vurderingsskjema eller fraværslogg.
- **md eller json-filer:** Brukes i utvikling og systemarbeid, men også egnet for strukturerte notater.
- **Andre tekster:** Du kan også kopiere tekst fra nettsider eller dokumenter og lime det inn i et Word-dokument før du laster det opp.

Det viktige er ikke hvilket filformat du bruker, men at innholdet kan leses som vanlig tekst. Når teksten først er hentet inn i notatboka, kan du sende den videre til agenten. Da kan agenten oppsummere lange tekster, forklare vanskelige avsnitt, hente ut datoer, navn eller nøkkelord, og foreslå forbedringer eller forkortelser. Kort sagt: Du gir agenten mer å jobbe med og du slipper å skrive alt selv.

Nå skal du få se hvordan du laster opp en tekstfil, leser innholdet og lar agenten bruke det som grunnlag for svaret.

Hvordan laste opp en fil i Colab?

Når du kjører koden files.upload(), åpnes et lite opplastingsvindu i Colab. Der kan du velge en .txt-fil fra datamaskinen din. Når du har valgt filen, blir den automatisk lagret i Colab og kan brukes videre i koden.



TIPS! Bruk et vanlig Word-dokument, for eksempel en ukeplan, en artikkel eller et refleksjonsnotat. Det viktigste er at teksten står som vanlige avsnitt i dokumentet.

Det første du må gjøre er å klargjøre agenten slik at den kan lese word-fila di. Kjør denne koden:

```
!pip install python-docx
```

Deretter limer du koden under:

```
!pip install -q python-docx

from pathlib import Path
from google.colab import files
import docx, openai
from
```

⚠️ NB! Last ned koden i TXT-format

Klikk på lenken under over for å åpne en ren og uformatert tekstversjon du kan kopiere trygt inn i Colab.

KI-agent kodefiler/Dokumentagent

```
# Se
open

FILE

# La
if not FILE.exists():
    uploaded = files.upload()
    FILE.write_bytes(uploaded[next(iter(uploaded))])
```

Hva gjør denne koden?

Denne koden lar deg laste opp et Word-dokument og bruke innholdet som grunnlag for agentens svar. Først installerer du pakken python-docx, som gjør det mulig å lese .docx-filer i Colab. Deretter åpnes dokumentet, og all teksten hentes ut.

Prøv noe annet med samme dokument

I stedet for å be om oppsummering, kan du teste andre typer spørsmål. Her er noen forslag du kan bruke i stedet for Oppsummer denne teksten:

- Hva er de tre viktigste poengene i denne teksten?
- Foreslå tre refleksjonsspørsmål elevene kan jobbe med.
- Forklar teksten slik at en 13-åring kan forstå den.
- Lag en kort innledning til teksten.

Bare bytt ut brukerens melding i messages-listen, resten av koden er den samme. På den måten kan du bruke dokumentet på mange måter, uten å laste det opp på nytt.

Neste kapittel

I neste kapittel skal du kombinere det du har lært og bygge din egen nytteagent, en agent med navn, stil og en konkret oppgave. Nå begynner det å ligne noe!



SKAP EN NYTTEAGENT

Gi agenten en konkret rolle

Nå har du lært å lage en enkel agent, gi den stil, huske samtaler og bruke verktøy og dokumenter. Du har alle byggeklossene på plass. I dette kapittelet skal du sette dem sammen til én komplett agent, din egen nytteagent.

En nytteagent har en konkret rolle. Den hjelper deg med én bestemt type oppgave, og kan brukes flere ganger. Det kan være alt fra å foreslå aktiviteter til å oppsummere møter eller bearbeide tekster. Du bestemmer selv hva agenten skal være god på og hvordan den skal oppføre seg. Her lærer du hvordan du definerer oppgave, tone og evner, og gir agenten et navn og en personlighet.

Hvorfor er dette viktig?

Når du lager en nytteagent, handler det ikke bare om å teste meldinger, du lager et lite verktøy som faktisk skal hjelpe deg med noe. Da må du være tydelig på hva agenten skal gjøre, hvordan den skal svare, og hvilke data eller tekster den skal bruke.

Jo mer presist du definerer rollen og tonen i starten, jo mer treffsikre og nyttige blir svarene du får tilbake. Det gjør agenten mer forutsigbar og enklere å bruke igjen senere.



TIPS! Tenk igjennom disse punktene først:

- Hva skal agenten hjelpe deg med? Skal den foreslå aktiviteter, skrive møtereferat eller hjelpe med tekstarbeid?
- Hvordan skal den svare? Skal tonen være formell, vennlig, ungdommelig eller rett-på-sak?
- Hva slags informasjon trenger den? Skal du gi den dokumenter, stikkord eller bare et åpent spørsmål?
- Hva skal agenten hete? (valgfritt) Å gi agenten et navn gjør det lettere å bruke den igjen og huske hva den gjør.

Eksempel

Vi starter med en enkel, gjenbrukbar agent som hjelper deg med å skrive forslag til ukeplan i norskfaget. Du kan bruke samme agent hver gang og bare endre hvilken uke eller hvilket tema du skriver inn.

Denne agenten:

- har en tydelig oppgave
- har en bestemt tone
- kan gjenbrukes med nye spørsmål
- er laget for å spare deg tid

Lim inn følgende kode hvis du har startet et nytt prosjekt. Hvis du fortsetter fra forrige kapittel, trenger du ikke.

```
!pip install python-docx
```

Deretter limer du inn koden under.

```
from google.colab import files
import docx, openai
from google.colab import userdata

# H
ope

# L
upl
fil

# L
doc = docx.Document(filnavn)
tekst = "\n".join([avsnitt.text for avsnitt in doc.paragraphs])
```

> ⚠️ NB! Last ned koden i TXT-format
>
> Klikk på lenken under over for å åpne en ren og uformatert tekstversjon du kan kopiere trygt inn i Colab.
>
> KI-agent kodefiler/Nytteagent

Hva gjør denne koden?

Denne koden lar deg laste opp et dokument med tekst fra forrige uke og bruke det som grunnlag for neste undervisningsforslag. Agenten leser dokumentet og foreslår tema, tekst og vurderingsform basert på innholdet. Det gjør agenten mer relevant og nyttig, og du sparer tid.

Neste kapittel

Du har nå bygd en fullt fungerende nytteagent som kan tolke tekst, hente informasjon og svare med stil. Det i seg selv er et stort steg, men det stopper ikke der.

I neste kapittel bygger vi videre på dette. Vi ser på hvordan du kan lage flere ulike agenter for ulike formål, med ulike roller og samtalestil, og hvordan du kan bruke dem i undervisning, refleksjon og lek.



TA ETT SKRITT VIDERE

Bygg mer avanserte KI-agenter

Nå har du lært hvordan man bygger en nyttig, samtalebasert agent med stil og verktøy. Men hva om du vil bruke dette til enda flere formål? Eller la elevene utforske mer på egen hånd?

I dette kapitlet får du ti forslag til videreutvikling. Alle agentene er ferdige og klare til bruk, du trenger bare å åpne dem i Colab og kjøre.

Skrivefeilagent

Du limer inn en tekst og agenten gir deg tilbakemelding på skrivefeil og forbedringer. Et godt eksempel på en enkel, men nyttig enveis-agent.

KI-agent kodefiler/Skrivefeilagent

Oppsummeringsagent

Du limer inn en lang tekst, og agenten lager en punktvis oppsummering. Kan brukes i alle fag.

KI-agent kodefiler/Oppsummeringsagent

Quizagent

Du limer inn en fagtekst, og agenten lager tre flervalgsoppgaver. Svært nyttig i undervisning eller som lekse.

KI-agent kodefiler/Quizagent

Tidstyvagent

Her føres en samtale. Du forteller hva du har brukt tiden på, og agenten hjelper deg med å identifisere "tidstyver" og foreslå forbedringer.

KI-agent kodefiler/Tidstyvagent

Etikkagent - interaktiv

Agenten hjelper deg med å tenke gjennom etiske dilemmaer. Den stiller spørsmål tilbake og bidrar til refleksjon. Fin i samfunnsfag og KRLE.

KI-agent kodefiler/Etikkagent

Rolleagent - interaktiv

Agenten hjelper deg med å tenke gjennom etiske dilemmaer. Den stiller spørsmål tilbake og bidrar til refleksjon. Fin i samfunnsfag og KRLE.

KI-agent kodefiler/Rolleagent

Planleggeragent

Du skriver inn fag og ukemål, og agenten lager et forslag til ukeplan. Nyttig for både elever og lærere.

KI-agent kodefiler/Planleggeragent

Spillagent - interaktivt

Agenten leder et lite tekstbasert eventyrspill. Den beskriver situasjoner, gir deg valg og reagerer på valgene dine.

KI-agent kodefiler/Spillagent

Spørsmålsagent - interaktivt

Du limer inn en fagtekst og stiller spørsmål om den. Agenten svarer tydelig og holder på samtalen.

KI-agent kodefiler/Spørsmålsagent

Forklaringsagent - interaktivt

Du limer inn vanskelige begreper eller tekstutdrag, og agenten forklarer dem på enkel norsk. Fin for elever som trenger språkstøtte.

KI-agent kodefiler/Forklaringsagent

TIPS! Hva mer kan du gjøre?

- Du kan be elevene lage egne agenter som svarer som en historisk person, en karakter fra en bok, eller som en bestemt yrkesrolle, f.eks. en advokat, psykolog eller personlig trener. Det gir både kreativitet og faglig fordypning.
- Dette fungerer spesielt godt når elevene selv får bestemme rollen, og lærer å styre stilen til agenten med systemmeldingen.

Neste kapittel

I dette kapitlet har du sett hvor langt du kan komme bare med samtale og instruksjoner. Noen agenter gjør én enkel oppgave. Andre holder en hel samtale.

I neste kapittel går vi et steg videre: Du skal lære hvordan en agent ikke bare snakker, men faktisk utfører en oppgave. Det gjør vi ved å gi agenten tilgang til noe som kalles et funksjonskall.



FUNKSJONSKALL

Gi agenten en funksjon

Så langt har du laget agenter som svarer på spørsmål, fører samtaler og har en bestemt rolle eller stil. Nå skal du lære hvordan du kan la agenten be programmet om å gjøre noe for seg, f.eks. regne ut noe, hente informasjon, eller utføre en handling. Dette kalles et funksjonskall.

Tenk på funksjonskall som et verktøyskrin du gir til agenten. Når brukeren spør om noe praktisk, som å regne ut et tall, hente været, eller slå opp et ord, forstår agenten at den trenger hjelp. Da finner den riktig verktøy i skrinet og bruker det automatisk i svaret sitt.

TIPS! Funksjonskall kan brukes til å

- bruke funksjonskall til å bygge agenter som henter vær, regner ut noe, leser filer eller snakker med andre API-er.
- Start med én funksjon, og test om agenten skjønner når den skal bruke den. Det er utrolig hvor mye det gir.

Eksempel: regneagentt

Vi skal lage en agent som kan regne ut svaret på et spørsmål som "Hva blir 13 + 27?". I stedet for å svare med gjetting, bruker agenten en faktisk Python-funksjon.

Steg 1: Lag funksjonen

Steg 2: Gi agenten beskjed om funksjonen
Vi forteller modellen at den kan bruke summer. Det gjør vi med en funksjonsbeskrivelse som forklarer:

- navn
- input
- forklaring

Steg 3: La agenten bruke funksjonen
Her bruker vi tool_choice="auto" - så modellen får velge om den vil bruke funksjonen.

```
import openai
from google.colab import userdata

openai.api_key = userdata.get("OPENAI_API_KEY")

messa
    {
treng
    }
]

respo
```

⚠️ NB! Last ned koden i TXT-format

Klikk på lenken under over for å åpne en ren og uformatert tekstversjon du kan kopiere trygt inn i Colab.

KI-agent kodefiler/Funksjonskallagent

Under følger tre andre agenter som bruker funksjonsoppkall. Last ned TXT-filene og kjør i Colab.

Jobbe med dato og tid
Regn ut hvor mange dager det er igjen til en innlevering.

```
import openai
from
opena
messa
    {
treng
    {
]
```

⚠️ NB! Last ned koden i TXT-format

Klikk på lenken under over for å åpne en ren og uformatert tekstversjon du kan kopiere trygt inn i Colab.

KI-agent kodefiler/Datoagent

Kalkulere poeng eller karakter
Beregn snittkarakter fra en liste med tall.

```
import openai
from
opena
messa
    {
treng
    {
]
```

⚠️ NB! Last ned koden i TXT-format

Klikk på lenken under over for å åpne en ren og uformatert tekstversjon du kan kopiere trygt inn i Colab.

KI-agent kodefiler/Snittberegningsagent

Lage forslag basert på tall
Brukeren skriver hvor mange timer de har sovet, trent og vært på skjerm og agenten gir et forslag til balanse.

```
import openai
from
opena
messa
    {
treng
    {
]
```

⚠️ NB! Last ned koden i TXT-format

Klikk på lenken under over for å åpne en ren og uformatert tekstversjon du kan kopiere trygt inn i Colab.

KI-agent kodefiler/Dagsanalyseagent

Neste kapittel

Du har nå lært hvordan en KI-agent kan bruke verktøy: hente data, regne ut noe, eller slå opp informasjon med hjelp av funksjonskall. Dette gjør agenten langt mer aktiv og nyttig i praktiske oppgaver.

I neste kapittel ser vi på noe helt grunnleggende: hva du gjør når noe ikke virker. Vi går gjennom vanlige feilmeldinger og hvordan du løser dem raskt og enkelt.



FEILMELDINGER

Vanlige feilmeldinger i Colab

Når du jobber med KI-agentene i denne boka, kan det hende du støter på feilmeldinger. Ikke få panikk! De fleste feil har en enkel forklaring og en enda enklere løsning. Her går vi gjennom de vanligste.

1. ❌ KeyError: 'OPENAI_API_KEY'
Årsak: Koden prøver å hente API-nøkkelen, men finner den ikke i userdata.
Løsning: Gå til menyen i Colab: ⚙️ (Settings) → Secrets
Sjekk at du har lagt inn en hemmelighet som heter OPENAI_API_KEY
Lim inn nøkkelen på nytt hvis du er usikker

2. ❌ ModuleNotFoundError: No module named 'docx'
eller No module named 'wikipedia'
Årsak: Koden prøver å bruke en pakke du ikke har installert ennå.
Løsning: Legg inn denne linjen øverst i koden og kjør den først:
python
KopierRediger
!pip install python-docx wikipedia

3. ❌ AttributeError: 'dict' object has no attribute 'tool_calls'
Årsak: Du har brukt feil model-parameter (gpt-3.5-turbo i stedet for gpt-4o) – og modellen forstår ikke funksjonskall.
Løsning: Sjekk at koden bruker riktig modell:
python
KopierRediger
model="gpt-4o"

4. ❌ NameError: name 'messages' is not defined
Årsak: Du bruker messages uten å ha laget listen først.
Løsning: Sørg for at du har denne linjen før du bruker messages:
python
KopierRediger
messages = [{"role": "system", "content": "..."}]

5. ❌ JSONDecodeError eller json.decoder.JSONDecodeError
Årsak: Modellen prøver å sende data til funksjonen, men formatet er litt feil – f.eks. hvis argumentene ikke er ordentlig formattert.
Løsning: Sjekk at du bruker: python
KopierRediger
import json
data = json.loads(tool_call.function.arguments)
Og at tool_choice="auto" er med i chat_completions.create(...)

6. ❌ SyntaxError eller IndentationError
Årsak: Noe i koden er feil formattert – ofte på grunn av feil innliming fra PDF eller nettside.
Løsning: Sjekk at alle innrykk er med (bruk fire mellomrom eller tab)

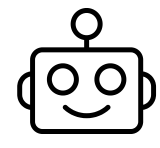


TIPS! Vær nøye på kodekjøring

- Bruk alltid .txt-filene som følger med boka. Å kopiere kode direkte fra PDF kan føre til små feil i koden.
- Kjør koden trinnvis og ikke alt på én gang. Da finner du raskt ut hvor feilen skjer.

Neste kapittel

Nå har du sett hvordan du håndterer de vanligste feilmeldingene, og hvordan du kan løse dem uten å miste motet. I neste kapittel handler det ikke om hva agenten gjør, men om hva du gjør. Vi skal se nærmere på ansvar, begrensninger og hva du bør tenke på når du bruker kunstig intelligens både som lærer og utvikler.





TA ANSVAR OVER AGENTEN

Kjenn til begrensningene

Nå har du lært å lage en agent som faktisk hjelper deg med forslag, oppsummering, planlegging og tilpasning. Du har sikkert begynt å se hvor nyttig dette kan bli. Men før du bruker agenten i det virkelige arbeidet ditt, må vi ta et viktig steg tilbake:

En KI-agent kan virke smart, men den forstår ikke innholdet, har ingen kontroll på fakta, og tar ikke ansvar for det den foreslår. Det må du gjøre.

Hva betyr det i praksis?

Det betyr at du alltid må vurdere om det agenten foreslår faktisk er riktig og relevant. Den kan tolke deg feil, finne på ting eller gi svar som ikke passer i konteksten. Du må også være sikker på at bruken er lovlig. Det er ikke tillatt å legge inn elevdata, personopplysninger eller sensitiv informasjon, og det gjelder også hvis agenten «bare» skal hjelpe deg med å bearbeide noe. Alt du gir til agenten, bør være åpent og ufarlig å dele.

Til slutt må du tenke over hva du faktisk gir fra deg. Hvis du for eksempel limer inn utdrag fra vurderingssamtaler eller logger, er du kanskje i strid med både personvern og skolens retningslinjer selv om det er ment som hjelp. Her er noen eksempler på hva du ikke bør gjøre:

- Ikke last opp vurderingssamtaler, elevtekster eller fraværslogg
- Ikke bruk agenten til å “finne ut” hvem som strever i klassen
- Ikke la agenten skrive individuelle tilbakemeldinger uten at du godkjenner det

TIPS! Dette kan du trygt bruke agenten til:

- Lage forslag til undervisning og ukeplaner
- Bearbeide åpne fagtekster
- Oppsummere artikler eller møteinnhold
- Skrive utkast til e-poster eller nyhetsbrev
- Foreslå aktiviteter eller refleksjonsspørsmål

Så lenge du holder på dette nivået, er det ingen fare. Spør deg selv: Hvis du ikke ville sendt det på e-post til en kollega uten å spørre først ikke legg det inn i agenten. Bruk åpne tekster, offentlige dokumenter og innhold du selv har laget.

Neste kapittel

I neste og siste kapittel får du tips til hvordan du kan bygge videre på agenten din. Vi ser på hvordan andre har brukt KI-verktøy på kreative måter og hva du kan utforske hvis du vil ta agenten et steg videre.



NESTE STEG VIDERE

Utforske videre selv

Nå har du laget din egen KI-agent, fra enkel samtale til nytteverktøy med stil, historikk og verktøy. Du har lært hvordan agenten fungerer, hva den kan gjøre, og hva den ikke skal gjøre.

I dette siste kapittelet får du inspirasjon og ideer til hvordan du kan utvikle agenten videre. Du trenger ikke gjøre noe mer for å bruke det du har lært, men hvis du vil utforske mer, finnes det mange muligheter.

Hva har andre gjort?

Mange lærere har allerede tatt i bruk enkle KI-agenter i hverdagen. Noen bruker dem til å forberede foreldremøter, andre til å oppsummere fagtekster eller finne forslag til årsplaner. Flere har laget egne agenter som hjelper dem med å bearbeide notater, forenkle dokumentasjon eller hente ut nøkkelpunkter fra ulike tekster. Felles for dem alle er at agenten tilpasses behovet og at brukeren har kontroll.



TIPS! Her er noen forslag til videre utbygging:

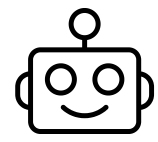
- **Bruk agenten sammen med elevene.** Lag en variant som forklarer fagstoff på ulike nivåer – f.eks. «Forklar denne teksten for en 8.-klassing» eller «Hva betyr dette avsnittet med enklere ord?»
- **Lag en "pedagogisk sparringspartner"**. En agent du kan stille spørsmål som: «Hva er fordeler og ulemper med å starte norsktimen med fri skriving?» eller «Hvilke grep kan gjøre vurderingssituasjonen mindre stressende?»
- **Eksperimenter med flerstegsoppgaver**. Be agenten først oppsummere en tekst, så foreslå oppgaver basert på den, og deretter lage en vurderingsrubrikk i én og samme samtale.
- **Bygg agenten inn i en nettside eller app.** Hvis du vil lære mer, kan du bruke verktøy som Streamlit eller Gradio til å gi agenten et brukergrensesnitt. Da kan andre bruke den også.
- **Test hvordan agenten takler ulike roller.** Lag en versjon som gir respons som en sensor, en kritisk leser, en elev, en foresatt – eller deg selv.

Hva har du lært?

Denne boka har tatt deg gjennom det viktigste du trenger for å bygge og forstå en enkel KI-agent. Du har lært å skrive meldinger, velge stil, bruke verktøy og jobbe med dokumenter og du har fått innsikt i hvordan agenten egentlig fungerer.

Målet har ikke vært å gjøre deg til programmerer. Målet har vært å gjøre deg trygg. Trygg på hva KI kan være, og trygg nok til å bruke det bevisst, ansvarlig og kreativt. Du trenger ikke kunne alt for å starte. Du trenger bare å begynne. Og nå har du gjort nettopp det.

Lykke til videre :)





NØKKELORD OG KILDER

Forklaringer på begreper brukt i boka

Colab (Google Colaboratory):
Et nettbasert Python-verktøy der du kan skrive, teste og kjøre kode gratis.

KI-agent (AI agent):
Et system som bruker språkmodeller og instruksjoner til å utføre oppgaver, svare, og handle på bakgrunn av innhold du gir.

OpenAI API:
En måte å sende spørsmål og hente svar fra GPT-modeller gjennom kode.

Personvern og ansvar:
Etiske og juridiske hensyn knyttet til bruk av KI, spesielt i skole og offentlig sektor. Du må selv kontrollere at bruk er i tråd med regelverk.

Python:
Programmeringsspråket som brukes til å gi agenten ekstra funksjonalitet.

Samtalehistorikk (messages):
En liste over meldinger som simulerer en samtale, med roller som system, user og assistant.

Språkmodell:
En algoritme som genererer tekst basert på sannsynlighet. Eksempel: GPT-4.

Systembeskjed (system prompt):
En instruksjon som bestemmer hvordan agenten skal oppføre seg – f.eks. tone, stil og rolle.

Verktøy:
Eksterne funksjoner som brukes sammen med språkmodellen, som værdata, filinnhold eller kalkulasjoner i Python.

Fine-tuning:
En metode for å trene modellen videre på spesialiserte data. Krever teknisk kompetanse og brukes i mer avanserte prosjekter.

GPT (Generative Pre-trained Transformer):
Modelltypen bak ChatGPT. Den er trent på store tekstmengder for å forutsi hva som kommer etter en tekst.

Hallusinasjon:
Når modellen finner på noe som virker riktig, men ikke stemmer. Dette er en kjent svakhet i alle språkmodeller.

LLM (Large Language Model):
En stor språkmodell, slik som GPT-4. Brukes til å generere tekst og svare på spørsmål.

Prompt:
Teksten du sender til modellen for å få et svar. Alt fra “lag en ukeplan” til “forklar denne teksten med enklere ord”.

Prompt engineering:
Å eksperimentere med hvordan du formulerer meldinger for å få bedre og mer presise svar.

RAG (Retrieval-Augmented Generation):
En teknikk der agenten henter inn tekster fra dokumenter eller databaser og bruker det som kunnskapsgrunnlag i svaret.

Token:
Et tekstfragment som språkmodellen bruker til å analysere og generere språk. En setning består ofte av 10–30 tokens.

Hva er en KI-agent – og hva kan den brukes til?
En grundig og oppdatert forklaring på hva KI-agenter er, hvordan de fungerer, og hvilke muligheter og utfordringer de gir. Inkluderer intervjuer med norske KI-eksperter: https://digitalnorway.com/aktuelt/hva-er-en-ki-agent-og-hva-kan-den-brukes-til

Introduksjon til kunstig intelligens – Tenk, Faktisk.no
En pedagogisk introduksjon til kunstig intelligens, med eksempler på KI-agenter i hverdagen og tips til kritisk tenkning:
https://tenk.faktisk.no/tema/kunstig-intelligens/introduksjon

Lag din egen generative AI: En enkel guide for nybegynnere. Praktisk guide for nybegynnere som vil forstå og lage egne KI-agenter. Forklarer generativ KI og hvordan slike agenter skapes og trenes:
https://vizologi.com/no/make-own-generative-ai-simple-guide-for-beginners/

Verktøy basert på kunstig intelligens i utdanning (UiB). Om bruk av KI-verktøy (som KI-agenter) i utdanning, og hvilke muligheter og utfordringer de gir: https://www.uib.no/foransatte/164914/verkt%C3%B8y-basert-p%C3%A5-kunstig-intelligens-i-utdanning-ansatte

KI på kontoret – alt du trenger for å komme i gang med KI på jobben. Bok (2025) som gir praktiske eksempler på hvordan KI-agenter kan brukes i arbeidslivet: https://www.ark.no/produkt/boker/fagboker/ki-pa-kontoret-9788234721946

KI replikerer vitenskaplig forskning innen KI. Artikkel om hvordan KI-agenter nå brukes til å reprodusere og forbedre vitenskapelig forskning, og hvilke konsekvenser dette kan få: https://www.datasor.no/paperbench-ki-replikerer-vitenskaplig-forskning-innen-ki/

OpenAI – What are AI Agents? OpenAI forklarer hva KI-agenter er, hvordan de brukes, og forskjellen på agenter og tradisjonell KI:
https://openai.com/blog/agents

Google DeepMind – Agents and Environments. DeepMind gir en oversikt over hvordan KI-agenter samhandler med miljøer og lærer gjennom erfaring: https://deepmind.com/research/publications/agents-and-environments

Stanford HAI – The Rise of AI Agents. Stanford Human-Centered Artificial Intelligence (HAI) diskuterer hvordan KI-agenter endrer teknologi og samfunn: https://hai.stanford.edu/news/rise-ai-agents

IBM – What are AI Agents? IBM gir en lettfattelig introduksjon til KI-agenter, med eksempler fra næringslivet: https://www.ibm.com/topics/ai-agents

© 2025 KENNETH BAREKSTEN

Denne boka er lisensiert under Creative Commons (CC BY 4.0). Det betyr at du kan dele, bruke og tilpasse innholdet, også kommersielt, så lenge du krediterer forfatteren.

Kontakt: kenneth@laererliv.no
Nettside: laererliv.no

LÆRERLIV
Fra klasserom til storsamfunn