Sensorveiledning TDT4114

Om emnet TDT4114 Anvendt Programmering

Emnebeskrivelse

Se emnebeskrivelse: https://www.ntnu.no/studier/emner/TDT4114#tab=omEmnet

Emnet dekker følgende læringsutbyttebeskrivelser i Anvendt Programmering:

- Grunnleggende programmeringsferdigheter, inkludert forståelse av variabler, datatyper, kontrollstrukturer (som løkker og betingede uttalelser), og funksjoner i Python.
- Grunnleggende konsepter i objektorientert programmering, som klasser, objekter, metoder, arv og polymorfi.
- Filbehandling og feilhåndtering i Python, inkludert hvordan man leser fra og skriver til filer, og hvordan man håndterer unntak.
- Dataanalyse og visualisering, inkludert bruk av bibliotekene NumPy, Matplotlib, SciPy og Pandas.
- Grunnleggende statistikk og sannsynlighetsregning, inkludert forståelse av konsepter som gjennomsnitt, median, standardavvik, korrelasjon, sannsynlighet og sannsynlighetsfordelinger.
- Lineær regresjon og andre prediktive modelleringsteknikker, inkludert hvordan man forbereder data for modellering, hvordan man trener og validerer en modell, og hvordan man tolker modellens resultater.
- Numeriske metoder, inkludert numerisk derivasjon og integrasjon, og løsning av likninger.
- Bruk av programmeringsmiljøer som Visual Studio Code og Jupyter Notebook, og versjonshåndtering med Git.
- Enhetstesting i Python, inkludert hvordan man skriver og kjører tester ved hjelp av unittest-rammeverket.

Lærestoff og verktøy

Tilgjengelige nettressurser via Blackboard og Jupyterboka.

Som IDE brukes Visual Studio Code.

Som versjonskontroll brukes GitHub/Gitlab. NTNU hoster egen GitHub (git.ntnu.no) som studentene kan bruke.

Om Mappen

I mappen inngår prosjektet miljødataanalyse og refleksjon over egenlæring.

Målsetning

I dette prosjektet skal dere utvikle en applikasjon som henter, analyserer og visualiserer miljødata fra åpne kilder, som værdata, luftkvalitetsdata, havnivådata etc. Prosjektet gir dere praktisk erfaring med programmeringskonsepter, dataanalyse og visualisering, samt innføring i versjonshåndtering og enhetstesting. Gjennom disse ferdighetene vil dere bli bedre rustet til å håndtere mer komplekse programmeringsoppgaver i fremtiden. Prosjektet bidrar ikke bare til å utvikle ferdigheter som er relevante for datavitenskap, men hever også den generelle kompetansen innen programvareutvikling.

Innledning og bakgrunn

I dagens samfunn er det en økende bevissthet rundt miljøspørsmål, drevet av bekymringer om klimaendringer, forurensning og bærekraftig utvikling. For å kunne ta informerte beslutninger og utvikle effektive strategier for å håndtere disse utfordringene, er det avgjørende å ha tilgang til og forståelse av miljødata. Miljødata kan gi innsikt i en rekke faktorer, inkludert værmønstre, luftkvalitet, vannkvalitet og økosystemers helse. Ved å analysere disse dataene kan vi identifisere trender, forutsi fremtidige forhold og utvikle tiltak for å forbedre miljøet. For mer informasjon om dette kan en studere f.eks. IPCC rapporter som gir vurdering av klimaendringer og deres konsekvenser.

Prosjektet med utvikling av en miljødataanalyseapplikasjon gir dere en unik mulighet til å jobbe med virkelige datasett fra åpne kilder, som meteorologiske institutter (f.eks. developer.yr.no) og miljøovervåkingsorganisasjoner (f.eks. https://www.eea.europa.eu/en/analysis). Dette gir ikke bare praktisk erfaring, men også en dypere forståelse av hvordan data kan brukes til å belyse komplekse miljøspørsmål. Gjennom prosjektet vil dere lære å navigere i ulike datakilder, vurdere datakvalitet og pålitelighet, samt anvende programmeringsferdigheter for å hente, bearbeide og analysere data.

I tillegg til å utvikle tekniske ferdigheter, vil prosjektet også fremme kritisk tenkning og problemløsning. Dere vil bli utfordret til å identifisere relevante datakilder, håndtere manglende verdier og uregelmessigheter i dataene, samt bruke statistiske metoder for å trekke meningsfulle konklusjoner. Dette er essensielle ferdigheter, spesielt i en tid der datadrevne beslutninger blir stadig mer sentrale i både offentlig og privat sektor. Videre vil prosjektet gi dere innsikt i viktigheten av versjonshåndtering og enhetstesting, som er kritiske komponenter i moderne programvareutvikling. Ved å bruke verktøy som Gitlab/GitHub for versjonering og unittest for testing, vil dere lære hvordan man sikrer kodekvalitet og samarbeider effektivt i team. Dette vil ikke bare heve deres tekniske kompetanse, men også forberede dere på kravene i arbeidsmarkedet, hvor samarbeid og kvalitetssikring er avgjørende.

Resultater

Ved prosjektets slutt får dere en fungerende applikasjon som dere leverer i Inspera for vurdering. Dette prosjektet er designet for å styrke deres programmeringsferdigheter, forståelse av dataanalyse og samarbeidskompetanse gjennom praktisk anvendelse.

Mappeoppgaver

Oppgave 1: Sett opp utviklingsmiljø

Før dere begynner med datainnsamlingen, er det viktig å sette opp et utviklingsmiljø som vil støtte dere gjennom hele prosjektet. Denne oppgaven vil veilede dere i å opprette et prosjekt i GitHub, installere nødvendige verktøy og opprette en test Jupyter Notebook for å sikre at alt fungerer som det skal.

Oppgave 2: Datainnsamling

I prosjektet for utvikling av en miljødataanalyseapplikasjon skal dere først identifisere relevante åpne datakilder, som f.eks. API-er fra meteorologiske institutter og miljøovervåkingsorganisasjoner. Deretter skal dere implementere funksjonalitet for å hente data fra disse kildene ved hjelp av Python-moduler (som eks. requests). For å integrere dataene i applikasjonen, bruker dere teknikker som håndtering av tekstfiler, CSV-filer, JSON-data, samt fil- og katalogadministrasjon. I tillegg skal dere benytte dere av list comprehensions, iteratorer og Pandas SQL (sqldf) for å utforske og forstå dataenes struktur og innhold før de forberedes for videre analyse.

Oppgave 4: Dataanalyse

I denne delen av prosjektet skal dere bruke verktøy som NumPy, Pandas, Matplotlib osv. til å beregne statistiske mål som gjennomsnitt, median og standardavvik, som er essensielle for å forstå datakvaliteten og identifisere trender. Videre vil dere implementere enkle statistiske analyser for å avdekke mønstre i dataene, noe som kan gi innsikt i miljøforholdene over tid. Denne analysen vil danne grunnlaget for videre visualisering og prediktiv analyse, og bidra til en dypere forståelse av de miljømessige faktorene som påvirker samfunnet.

Oppgave 5: Visualisering

Denne oppgaven fokuserer på bruken av Matplotlib (og Seaborn) for å skape informative og engasjerende visualiseringer av de analyserte miljødataene. I denne delen av prosjektet skal dere utvikle grafer og diagrammer som eksempelvis illustrerer endringer i luftkvalitet over tid, sammenligning av temperaturdata, og andre relevante trender. Målet er å gjøre dataene mer tilgjengelige og forståelige for et bredere publikum. I tillegg, dersom tiden tillater det, skal dere utforske muligheten for å lage interaktive visualiseringer ved hjelp av Widgets, Plotly eller Bokeh, noe som kan gi brukerne en mer dynamisk opplevelse av dataene.

Oppgave 6: Prediktiv analyse

Denne oppgaven handler om prediktiv analyse og fokuserer på implementeringen av lineær regresjon ved hjelp av scikit-learn for å forutsi fremtidige miljøforhold basert på historiske data. I denne delen av prosjektet skal dere forberede dataene ved å identifisere relevante funksjoner og målvariabler, samt håndtere eventuelle manglende verdier som kan påvirke modellens nøyaktighet. Deretter skal dere trene regresjonsmodellen på de rensede dataene, evaluere dens

ytelse ved hjelp av passende metoder som beregning av feilmål, og til slutt bruke modellen til å lage prediksjoner for fremtidige miljøforhold.

Oppgave 7: Refleksjonsnotat

Skriv et refleksjonsnotat (maks 800 ord) etter gjennomføringen av prosjektet. Denne skal inneholde viktige punkter som gir innsikt i deres læringsprosess, erfaringer og vurderinger av prosjektet.

Karakterskala ved NTNU

Karakterbeskrivelse for emner i teknologiske fag ved NTNU:

Symbol	Betegnelse	Generell, ikke fagspesifikk beskrivelse av vurderingskriterier
A	Fremragende	Kandidaten viser særdeles god kunnskap i og oversikt over emnets faglige grunnlag og innhold. Kandidaten viser meget stor grad av selvstendighet og solid analytisk forståelse. Kandidaten viser svært gode ferdigheter i anvendelsen av denne kunnskapen.
B	Meget god	Kandidaten viser meget god kunnskap i og oversikt over emnets faglige grunnlag og innhold. Kandidaten viser betydelig grad av selvstendighet og god analytisk forståelse. Kandidatens ferdigheter i anvendelsen av denne kunnskapen ligger over gjennomsnittet.
C	God	Kandidaten viser god oversikt over de viktigste kunnskapselementene og sammenhengene i emnets faglige grunnlag og innhold. Kandidaten viser selvstendighet. Kandidaten viser analytisk evne og forståelse. Kandidaten viser gjennomsnittlig evne til å anvende sin kunnskap. Gjennomsnittet kan dels tolkes som typisk prestasjon av mange studenter i emnet og dels som krav til tilfredsstillende god prestasjon i emnet.
D	Nokså god	Kandidaten viser i noen grad analytisk evne og forståelse. Kandidaten viser selvstendighet i noen grad. Kandidaten viser oversikt over de viktigste kunnskapselementene og sammenhengene i emnets faglige innhold, men denne oversikten er preget av noen vesentlige mangler. Kandidaten viser i noen grad evne til å bruke kunnskapen aktivt, men prestasjonen er noe dårligere enn gjennomsnittet.
E	Tilstrekkelig	Kandidaten viser mangelfull analytisk evne og forståelse. Kandidaten viser gjennomgående noe, men sporadisk preget oversikt over de viktigste kunnskapselementene og sammenhengene i emnets faglige innhold. Kandidatens prestasjon oppfyller minimumskravet som stilles i emnet når

		det gjelder kunnskap, analytisk evne og ferdighet i å anvende emnets kunnskapsinnhold.
F	Ikke bestått	Kandidatens prestasjon faller under minimumskravet som stilles i emnet når det gjelder kunnskap, analytisk evne og ferdighet i å anvende emnets kunnskapsinnhold.

Sensurskjema

Under følger sensurskjema som er lagt til grunn for sensureringen av mappene.
Poenger og vekting er ikke oppgitt i denne veiledningen. Det vil alltid gjøres en helhetsvurdering av kandidatens besvarelse før endelig karakter settes.

Vurderingskriterier - Mappe - TDT4114 Vår 2025				
				Score Prosjekt
KRITERIUM	**Ikke bestått (F)** **0 - 39p**	**Svak (E, D)** **40 - 59p**	**Middels (C)** **60 - 79**	**Utmerket (B, A)** **80 - 100p**
Utviklingsmiljø				
Mappestruktur: - Har README.md filer som beskriver innholdet i mappen - Har plassert datafiler i egne foldere - Har plassert notebookfilene i egne foldere - Har plassert python skripter i src folder - Har plassert testene i testsfolder - Har ressourser (bilder osv.) i egen folder - Har opprettet requirements.txt - Har releasenote - Har .gitignore	Har ikke - mappestruktur - .gitignore	Følger noenlunde anbefalt mappestruktur, men: - Har ikke test-klassene i egen test-mappe - Har ikke ressurser i resources-mappen - Har ikke .gitignore, releasenote	Har en god mappestruktur i hht til standarder nevnt i Jupyterboka	Har i tillegg - en logisk katalogstruktur for ressursfiler (f.eks. bildefiler/ikoner) i egen resources-mappe. - forskjellige datatype (excel, cvs, ..) plasseres i forskjellige data-foldere. - bruker virtuell miljø
Versjonskontroll				

Lokalt/sentralt repo, commits og branching	Har ikke benyttet versjonskontroll	- Prosjektet er lagt til versjonskontroll lokalt - Sporadiske innsjekkinger (commits) - mye som er endret mellom hver commit - Mangelfulle commit-meldinger (enten tomme, eller ikke beskrivende for endringene som er utført) - Ikke sentralt repo	- Prosjektet har sentralt repo (GitHub/GitLab) - Fornuftig jevnlig innsjekking (commit) av endringer - Gode commit-meldinger som beskriver kort hvilke endringer som er gjort/hvilke problem som er løst - OK om greiner (branches) ikke er benyttet	- Har benyttet greiner som del av arbeidsflyt (f.eks. develop/main), for features/utprøving og liknende. - Har gjennomført merge mellom greiner - Har benyttet tags for å merke versjoner
Filer lagt til versjonskontroll	Har ikke benyttet versjonskontroll	- Benytter IKKE .gitignore - Har lagt unødvendige filer/mapper til versjonskontroll, som venv, kompilerte filer (__pycache__), _build osv.	- Benytter .gitignore - Har filtrert bort de fleste filer og mapper	- Benytter .gitignore - Kun filer underlagt versjonskontroll som er lagt til (alle IDE-spesifikke filer er filtrert bort)
Enhetstesting				
	Har ingen enhetstester	- Har noen enhetstester/tester noen få av funksjonene/klassene - Mangler negative tester/har ikke forstått "negativ test" - Ikke gode beskrivende navn på testene	- Har gode beskrivende navn på testene - Har enhetstester for de grunnleggende funksjonene/klassene	- Har helt greie negative tester (viser at kandidaten har forstått hovedpoenget med positive/negative tester) - Følger prinsippet om Arrange-Act-Assert

Kodekvalitet				
Dokumentasjon	Ingen eller svært mangelfull dokumentasjon. Ingen eller småpussede kommentarer	Noe dokumentasjon med kommentarer, men ufullstendig eller inkonsekvent.	Fullstendig og konsekvent dokumentasjon av kode	- Klare funksjonsdokumentasjon (__doc__) på funksjoner/metoder
Kodestil (PEP8 eller tilsvarende standarder)	Ikke fulgt kodestandarder, inkonsekvent formatering, feil i innrykk, linjelengde osv.	Følger i hovedsak PEP8 (eller tilsvarende standard), men noen inkonsistenser finnes (f.eks. variabelnavn, linjelengde).	Strengt fulgt PEP8 (eller tilsvarende standrad), godt strukturert kode, konsekvent format og navngivning.	-Følger PEP8 eller tilsvarende standarder.
Navngivning	Variabler, funksjoner og klasser har vage eller utydelige navn som ikke gjenspeiler funksjon.	Bruker delvis beskrivende navn, men enkelte navn kan være kortfattede eller uklare.	I tillegg har filer og mapper gode navn	Alle variabler, funksjoner og klasser har beskrivende, men konsise navn som er lett forståelige. Følger PEP8 eller tilsvarende for navngiving av variabler og filnavn/mapper.
Robusthet	Ingen eller dårlig validering av parametre.	Svak eller ingen unntakshåndtering.	Enkel validering av parametre, unntak håndteres delvis (noen try/except).	Full god praksis på parameterverifisering, robust unntakshåndtering med relevante feilmeldinger.
Kodestruktur	Ingen struktur, mye hardkoding	Noe bruk av funksjoner/modulalisert kode	Delvis modulær, men kan være noe duplisering og uoversiktlig.	God modulær struktur, funksjoner/metoder følger single responsibility principle, minimalt med duplisering
Kommentarer	Ingen kommentarer	Lite kommentarer, spesielt ikke i komplekse deler.	Noen forklarende kommentarer, men ujevn kvalitet.	Gode informative kommentarer som forklarer hensikt og logikk, ikke overdreven

Kvaliteten på datainnsamlingen og forberedelsen				
Har identifisert relevante og pålitelige åpne datakilder	Har ikke brukt åpne kilder	Brukt åpne kilder, men ingen beskrivelse eller dokumentasjon	Relevante og pålitelige kilder, dokumentert delvis.	Godt dokumenterte, autoritative og relevante kilder.
Har identifisert og implementert teknikker for å hente data	Ingen teknikker for datainnhenting. Har hardkodet data i programmet	Har valgt teknikker for å lese csv, json, excel,… men har ikke sagt noe om datakvalitet og prosessen videre	Har valgt teknikker for å lese csv, json, excel,… har delvis sagt om kvaliteten og prosessen videre	Har i tillegg brukt API for å lese data og lagre til csv, json eller annen format
Har tatt i bruk teknikker for å utforske dataene	Har ikke brukt noen teknikker	Utforsker dataene uten å bruke biblioteker som pandas, numpy, etc.	Har brukt pandas eller andre biblioteker for å utforske data (f.eks.describe, info, head, sample, etc)	Bruker teknikker som pandas.query og eller bitvise operatorer for å filtrere data. Bruker pandasql til å utforske data
Har beskrevet alle dataene	Har ikke beskrevet dataene (kolonnene)	Har beskrevet delvis kolonnene og deres datatyper	Har beskrevet alle kolonnene og deres datatyper	Har i tillegg identifisert kolonner som ikke skal være med i resten av analysen. Har identifisert nye kolonner som må beregnes på grunnlag av eksisterende kolonner
Har håndtert manglende verdier	Ingen håndtering av manglende verdier. Har ikke konstruert manglende verdier for videre prosessering	Mangler ikke verdier i de innleste dataene, men har innført feil i dataene	Har håndtert alle manglende verdiene	Har i tillegg visualisert manglende verdier, tatt stilling til og begrunnet håndtering av manglende verdier
Har håndtert duplikater	Ingen håndtering av duplikater.	Har fjernet duplikater delvis	Har fjernet alle duplikatene	Har i tillegg definert hvilke kolonner som betraktes som duplikater

har brukt teknikker for å utforske/manipulere dataene	Studenten har ikke nevnt eller brukt noen teknikker for å utforske eller manipulere dataene. Ingen referanser til relevante biblioteker (som Pandas) eller programmeringsmetoder.	-har brukt enkle Pandas-operasjoner som .head(), .info(), eller .describe(). - ingen avanserte metoder eller iterasjoner. - begrenset forståelse av datautforskning og manipulasjon	- har brukt flere relevante teknikker for å utforske og manipulere dataene. -inkluderer bruk av Pandas-funksjoner som groupby(), .apply() -inkluderer enkle iteratorer, løkker, eller list comprehensions.	-inkluderer avanserte Pandas-operasjoner .merge(), .melt(), eller bruk av json_normalize. -bruker iteratorer, list comprehensions, eller tilpassede funksjoner for spesifikke oppgaver. -inkludere visualiseringer (f.eks. missingno for manglende verdier) eller bruk av SQL-lignende operasjoner med Pandas SQL (sqldf). -demonstrerer en dyp forståelse av datautforskning og manipulasjon.
Har beskrevet uregelmessigheter i dataene og håndtering av dem	Ingen uregelmessigheter nevnt eller plan for håndtering	Identifiserer uregelmessigheter (f.eks. manglende verdier eller duplikater) men forklarer ikke hvordan disse skal håndteres	Identifiserer uregelmessigheter (f.eks. manglende verdier eller duplikater) med enkle løsninger (f.eks. sletting).	Identifiserer flere relevante uregelmessigheter (f.eks. outliers, inkonsistente formater) og foreslår passende teknikker (f.eks. imputasjon, transformasjon).
Har brukt teknikker som maskering og utligger identifisering	Har ikke brukt maskering eller håndtert utligger verdier. Har ikke konstruert utliggerverdier hvis disse ikke originalt er med i dataene	Har håndtert utligger data, men har ikke konstruert nye dersom original data mangler utligger verder	Har konstruert og håndtert utliggerverdier (dersom de manglet)	Har i tillegg brukt maskering og beregnet statistiske mål som gjennomsnitt, median, stdv osv..
Har håndtert kategoriske data	Har ikke håndtert kategoriske data	Har delvis håndtert kategoriske data	Har håndtert alle kategoriske data	Har brukt flere teknikker (som f.eks. Label Encoding og OneHot Encoder)

Har identifisert avhenige og uavhengige variabler	Har ikke identifisert avhengige og uavhengige varibler. Har lite eller ingen forståelse av hvordan variablene relaterer til hverandre i en datamodell	Har identifisert uavhengige men ikke avhengige eller omvendt.	- har identifisert avhengige og uavhengige variabler. - kan tydelig definere og forklare avhengige og uavhengige variabler	- undersøker og vurderer hvordan variasjoner i en uavhengig variabel påvirker utfalle av en avhengig variabel - har definert flere uavhengige variabler
Dyktighet i dataanalyse og bruk av statistiske metoder				
Har brukt verktøy som NumPy og Pandas til å beregne statistiske mål	Ingen bruk av NumPy eller Pandas nevnt, og ingen forklaring på viktigheten av statistiske mål.	Viser enkel bruk av NumPy eller Pandas for å beregne gjennomsnitt, median eller standardavvik, men uten dypere forklaring på hvorfor de er viktige.	Demonstrerer korrekt bruk av både NumPy og Pandas for å beregne gjennomsnitt, median og standardavvik. Gir en grunnleggende forklaring på hvorfor disse målene er viktige (f.eks. for å forstå datasettet).	Demonstrerer korrekt bruk av både NumPy og Pandas for å beregne gjennomsnitt, median og standardavvik. Gir en grunnleggende forklaring på hvorfor disse målene er viktige (f.eks. for å forstå datasettet).

har undersøkt sammenhengen mellom to variabler i datasettet	Ingen eksempel eller forklaring på hvordan sammenhengen mellom to variabler kan analyseres.	Viser et enkelt eksempel, som å bruke Pandas til å beregne korrelasjon (.corr()), men uten videre forklaring eller tolkning.	Gir et konkret eksempel, som å bruke scatterplot for visualisering og korrelasjonskoeffisient for å kvantifisere sammenhengen. Forklarer kort hva resultatene betyr	Inkluderer både en statistisk metode (f.eks. korrelasjon eller regresjonsanalyse) og en visualisering (f.eks. scatterplot med trendlinje). Forklarer tydelig hvordan analysen er implementert og tolker resultatene i konteksten av datasettet
har håndtert skjevheter i dataene og forklart metoder for å håndtere dem	Ingen omtale av skjevheter eller metoder for å håndtere dem.	Nevner en metode for å håndtere skjevheter (f.eks. utligger), men uten videre forklaring eller vurdering av pålitelighet.	Identifiserer potensielle skjevheter (f.eks. outliers) og foreslår flere metoder for å håndtere dem (f.eks. fjerning av utligger).	Diskuterer kort hvordan dette bidrar til pålitelig analyse.
har identifisert hvilke visualiseringer støtter analysen	Ingen visualiseringer nevnt eller ingen forklaring på hvordan de kan støtte analysen.	Nevner enkle visualiseringer (f.eks. histogram eller scatterplot) uten videre forklaring på hvordan de hjelper med analysen eller formidlingen.	Beskriver flere relevante visualiseringer (f.eks. histogrammer, scatterplots, boxplots) og forklarer hvordan de støtter analysen (f.eks. identifisere mønstre, utliggere eller fordeling	Inkluderer et bredt spekter av visualiseringer (f.eks. heatmaps, linjediagrammer, korrelasjonsmatriser)
Kvaliteten og klarheten i visualiseringene				

har brukt matplotlib/seaborn til å visualisere dataene	Ingen omtale av Matplotlib eller Seaborn, eller ingen forslag til hvordan de kan brukes.	Enkle visualiseringer (f.eks. linjediagram eller histogram) uten videre forklaring på hvordan de forbedrer forståelsen av dataene.	Enkle visualiseringer (f.eks. linjediagram eller histogram) med forklaring på hvordan de forbedrer forståelsen av dataene.	Beskriver flere relevante visualiseringer (f.eks. scatterplots, heatmaps, boxplots) og forklarer hvordan de hjelper med å identifisere mønstre, trender eller utliggere i dataene. Nevner spesifikke funksjoner som plt.plot(), sns.heatmap(), eller sns.boxplot().
Har håndtert manglende verdier og visualiseringene	Ingen omtale av hvordan manglende data håndteres eller visualiseres.	Nevner enkel metode for håndtering (f.eks. fjerne rader med manglende data) eller visualisering (f.eks. en graf uten spesifikk behandling av manglende verdier).	Bruker metoder for å estimere manglende data (f.eks. interpolasjon, ffill, etc.)	I tillegg forklarer metodene som er brukt
har laget interaktive grafer	Ingen omtale av prosessen for å lage interaktive visualiseringer	Gir en enkel beskrivelse av hvordan et verktøy (f.eks. matplotlib.widgets) kan brukes til å lage interaktive visualiseringer.	implementerer i tillegg en slik graf	Inkluderer eksempler på funksjoner som interaktive akser, tilpassede verktøytips etc.
evaluerer effektiviteten av visualiseringene	Ingen omtale av hvordan visualiseringene evalueres eller hvordan de formidler funnene.	Gir en enkel vurdering, som å nevne om visualiseringene er "klare" eller "lesbare", uten spesifikke metoder for evaluering	Diskuterer hvordan visualiseringene hjelper med å formidle funnene (f.eks. ved å fremheve trender eller mønstre).	Inkluderer i tillegg kriterier som lesbarhet, estetikk…
Transformering av data				

Har håndtert kategoriske data	Har ikke håndtert kategoriske data	Har delvis håndtert kategoriske data	Har håndtert alle kategoriske data	Har brukt flere teknikker (som f.eks. Label Encoding og OneHot Encoder)
Deling av data i test og trening	Har ikke delt data i trening og test sett	Har delt data i test og trening manuelt (ikke brukt bibliotek)	Har f.eks. brukt train_test_split fra sklearn	Har forklart hensikten med inndeling av dataene i 2 datasett
Eskalering av data (feature scaling)	Har ikke normalisert eller standardisert data	Har normalisert/standardisert, men ingen forklaring	Har normalisert/standardisert data og begrunnet hvorfor	Gir en grundig forklaring av normalisering og standardisering, inkludert formler og kontekst for når de skal brukes
Modellering				
har implementert regresjonsmodell	Har ikke definert regresjonsmodell	Har implementert enkel linear regresjonsmodell fra sklearn (LinearRegression)	Bruker i tillegg modellen til å prediktere	Evaluerer modellens ytelse f.eks. ved hjelp av MSE eller R2.
Dokumentasjon/Refleksjon				
Struktur	Ingen struktur i beskrivelsene	Litt forklaringer i programkode, notebooks og markdownfiler	Har forklaring og beskrivelser på det meste.	Bruker README.md filer for overordnede beskriveler og forklaringer, brukt notebooks for å drøfte resultater og forklare oppgaver som løses. Har har kodedokumentasjon i programfiler (*.py)
Refleksjoner over læring	Ingen refleksjon over læring eller utvikling.	Enkel refleksjon, nevner noen få læringspunkter uten dybde.	Reflekterer over flere læringspunkter, inkludert bruk av verktøy og konsepter.	Dyp refleksjon med konkrete eksempler på læring og hvordan det kan anvendes i fremtiden.
Identifisering av utfordringer	Ingen utfordringer nevnt.	Nevner én eller to utfordringer uten detaljer eller løsninger.	Identifiserer flere utfordringer og beskriver hvordan de ble håndtert.	Identifiserer utfordringer med detaljerte beskrivelser av løsninger og refleksjon over læring.

Refleksjoner over samarbeid	Ingen refleksjon over samarbeid.	Enkel refleksjon, nevner fordeling av oppgaver uten vurdering av effektivitet.	Reflekterer over samarbeid, inkludert kommunikasjon og oppgavefordeling.	Dyp refleksjon over samarbeid med vurdering av styrker, svakheter og forbedringsmuligheter.
Vurdering av resultater	Ingen vurdering av resultater.	Enkel vurdering, nevner kvaliteten på visualiseringer eller analyser uten detaljer.	Vurderer kvaliteten på resultater med eksempler og forslag til forbedringer.	Utførlig vurdering av resultater med konkrete eksempler og refleksjon over hva som fungerte godt.
Ideer til forbedringer	Ingen ideer til forbedringer nevnt.	Nevner én eller to forbedringer uten detaljer.	Beskriver flere forbedringer	Utførlig beskrivelse av forbedringer med refleksjon over hvordan de kan implementeres
Oppsummering av læringspunkter	Ingen oppsummering av læringspunkter.	Enkel oppsummering uten fokus på de viktigste læringspunktene.	Oppsummerer de viktigste læringspunktene med eksempler.	Utførlig oppsummering med refleksjon over hvordan læringen bidrar til forståelse av datavitenskap.
Personlige tanker om fremtidig anvendelse	Ingen personlige tanker nevnt.	Nevner én eller to anvendelser uten refleksjon.	Reflekterer over flere måter erfaringene kan anvendes i fremtidige studier eller yrkesliv.	Utførlig refleksjon over hvordan erfaringene kan anvendes, med konkrete eksempler og relevans.