Eksamen, IN2110, vår 2020

Bakgrunn

- Vi anbefaler å lese gjennom hele oppgaveteksten (4 sider) før du begynner. Hvis du føler du savner informasjon for å løse en oppgave, gjør dine egne antakelser og redegjør for dem.
- De mulige poengene angitt for hver seksjon summerer til 100.
- Svarene dine må leveres som én enkelt PDF-fil. Dersom en oppgave ber om en graf, et diagram eller annen type visualisering så må dette inkluderes i PDF'en.
- Der vi oppgir omtrentlig forvented lengde på svar tar vi utgangspunkt i skriftstørrelse 12, og 1.5 linjeavstand.

1 Representasjon av ord (20 poeng)

(a) Beskriv kort hva som menes med 'den distribusjonelle hypotesen'. (Bruk maksimalt en halv side.)

(b) Forklar kort (maksimalt en halv side) hvordan vi kan implementere den distribusjonelle hypotesen i en vektorrom-modell.

(c) Skisser kort noen mulige problemer eller utfordringer med å brukes en vektor-basert distribusjonell tilnærming for å representere ords betydning. (Bruk maksimalt en halv side).

(d) Én type vektorrepresentasjoner vi snakket om i forelesningene er såkalte *word embeddings*. Forklar kort (ett avsnitt) hva som kjennetegner disse.

(e) Her skal vi jobbe med pre-prosessering av tekst. Ta utgangspunkt i følgende tekst som eksempel:

The window washers washed the windows.

Forklar og vis med utgangspunkt i eksemplet over hva vi mener med tokenisering, lemmatisering og stemming. Ved telling av ordforekomster har vi gjerne skilt mellom to nivåer; *types* (ordtyper) og *tokens* (enkeltord). Oppgi det totale antall typer og tokens for hvert trinn; rå tekst, tokenisert, lemmatisert og til slutt med stemming. Kan du komme på andre former for normalisering av teksten som kan være aktuelle? Diskuter i så fall effekten av disse også.

2 Klassifikasjon (17 poeng)

(a) Forklar kort forskjellen på *klassifikasjon* og *klyngeanalyse* (*clustering*). Prøv å ta i bruk relevant fagterminologi. Forklar også kort hvilke fordeler og ulemper du ser med disse to ulike tilnærmingene? (Bruk ca. et avsnitt totalt.)

(b) Beskriv kort metodene logistisk regresjon og kNN for klassifikasjon. Sammenlikne egenskapene deres og beskriv kort deres fordeler og ulemper. (Bruk maksimalt en halv side.)

(c) I emnet har vi snakket om ulike mål for å evaluere ytelsen til en klassifikator. To av målene vi har snakket om er *Precision* og *Recall*. Disse to målene er til en viss grad komplementære og i en del situasjoner vil det kunne finnes en avveining (*trade-off*) mellom å oppnå en høy verdi for enten *Precision* eller *Recall*. Forklar hva vi mener med dette. Beskriv også en vanlig måte for å kombinerer de to målene til ett mål. (Bruk maksimalt en halv side.)

3 Logistisk regresjon (15 poeng)

Last ned filen `https://github.uio.no/IN2110/in2110-lab/blob/master/ekstra/insekter_fra_Ecuador.csv`, som inneholder en tabell med fire kolonner. Den første kolonnen henviser til navn på

wikipedia-sider, mens den andre og tredje kolonnen indikerer henholdsvis hvor mange tokens og hvor mange lenker det finnes på denne siden. Den fjerde kolonnen er en binær variabel som indikerer om siden handler om et insekt fra Ecuador.

Du kan enkelt konvertere filen til en Pandas DataFrame og dele den i et treningsett og et testsett som følger:

```
>> df = pandas.read_csv("insekter_fra_Ecuador.csv", index_col=0)
>> train_df, test_df = sklearn.model_selection.train_test_split(df, shuffle=False)
```

(a) Tren en logistisk regresjonsmodell på `train_df` (uten regularisering) som predikerer om wiki-siden handler om et insekt fra Ecuador basert på to numeriske trekk (*features*), nemlig antall tokens og antall lenker på siden. Hvilke *accuracy* oppnår du på testsettet `test_df`?

(b) Forklar hvorfor verdien du oppnådde for *accuracy* er så lav, basert på det du vet om logistisk regresjon. For å støtte din forklaring, tegn en *scatter plot* (du kan bruke funksjonen `scatterplot` i Seaborn[1]) hvor de to aksene står for de to numeriske trekkene, og fargen (*hue*) representerer outputklassen (insekt fra Ecuador eller ikke). Tegn deretter beslutningslinjen som er assosiert med den logistiske regresjonsmodellen du nettopp har trent.

Bruk maksimalt én side for spørsmålene ovenfor, herunder både forklaringene dine og figuren med *scatter plot*. Du trenger ikke å levere noen kode.

4 Sekvensmodeller (18 poeng)

Last ned filen `https://github.uio.no/IN2110/in2110-lab/blob/master/ekstra/norne.txt` som inneholder tokeniserte setninger fra NorNE-korpuset, med en linje per setning. Hver token er etterfulgt av en understrek og ordklassen.

La oss nå anta at vi er gitt den følgende ordsekvensen: "*flere taler*". Vi ønsker å finne ut om den meste sannsynlige ordklassen for ordet "*taler*" er NOUN eller VERB, gitt at "*flere*" har ADJ som ordklasse.

(a) Gitt setningene i filen `norne.txt`, estimer de to transisjonssansynnlighetene $P(s_1 = \mathsf{NOUN}|s_0 = \mathsf{ADJ})$ og $P(s_1 = \mathsf{VERB}|s_0 = \mathsf{ADJ})$.

(b) Gitt setningene i filen `norne.txt`, estimer de to emisjonsssansynnlighetene $P(o_1 = \textit{taler}|s_1 = \mathsf{NOUN})$ og $P(o_1 = \textit{taler}|s_1 = \mathsf{VERB})$. Dere kan ignorere smoothing.

(c) Gitt sansynnlighetene dere nettopp har beregnet, hva er den mest sannsynlige ordklassen for ordet "*taler*" som kommer etter "*flere*"?

Bruk maksimalt én side for spørsmålene ovenfor, inkludert beregningene dine og begrunnelsen bak dem. Du trenger ikke å levere noen kode.

[1] `https://seaborn.pydata.org/generated/seaborn.scatterplot.html`

5 Dependenssyntaks og parsing (15 poeng)

Step	Stack	Word list	Action
0	[root]	[The, cat, could, have, chased, those, dogs, down, the, street]	SHIFT
1	[root, The]	[cat, could, have, chased, those, dogs, down, the, street]	LEFT-ARC$_{det}$
2	[root]	[cat, could, have, chased, those, dogs, down, the, street]	SHIFT
3	[root, cat]	[could, have, chased, those, dogs, down, the, street]	SHIFT
4	[root, cat, could]	[have, chased, those, dogs, down, the, street]	SHIFT
5	[root, cat, could, have]	[chased, those, dogs, down, the, street]	LEFT-ARC$_{aux}$
6	[root, cat, could]	[chased, those, dogs, down, the, street]	LEFT-ARC$_{aux}$
7	[root, cat]	[chased, those, dogs, down, the, street]	LEFT-ARC$_{nsubj}$
8	[root]	[chased, those, dogs, down, the, street]	RIGHT-ARC$_{root}$
9	[root, chased]	[those, dogs, down, the, street]	SHIFT
10	[root, chased, those]	[dogs, down, the, street]	LEFT-ARC$_{det}$
11	[root, chased]	[dogs, down, the, street]	RIGHT-ARC$_{dobj}$
12	[root, chased, dogs]	[down, the, street]	SHIFT
13	[root, chased, dogs, down]	[the, street]	SHIFT
14	[root, chased, dogs, down, the]	[street]	LEFT-ARC$_{det}$
15	[root, chased, dogs, down]	[street]	LEFT-ARC$_{case}$
15	[root, chased, dogs]	[street]	REDUCE
16	[root, chased]	[street]	RIGHT-ARC$_{obl}$
17	[root, chased, street]	[]	REDUCE
18	[root, chased]	[]	REDUCE
19	[root]	[]	DONE

(a) I tabellen over ser du en transisjonssekvens for en dependensparser.

(i) Tegn dependensgrafen som parseren produserer.

(ii) Hvilken parsingsalgoritme (arc standard eller arc eager) er brukt her? Begrunn svaret ditt.

(b) Velg deg ut tre formelle kriterier som ofte stilles til dependensgrafer og gi en kortfattet forklaring for hver av dem. Vis deretter hvordan du kan endre dependensgrafen fra (a) slik at den bryter minst to av disse. Du kan her velge om du vil tegne de modifiserte grafene eller forklare endringene med ord.

(c) Hva slags trekk bruker man typisk for klassifisering av transisjoner i dependensparsing? Illustrer svaret ditt ved å gi eksempler på minst tre typer trekk sammen med verdiene de vil få i steg 12 i transisjonssekvensen over.

Bruk maksimalt én side for spørsmålene ovenfor, herunder både grafen(e) og forklaringene dine.

6 Interaktive systemer (15 poeng)

La oss anta du ønsker å utvikle et dialogsystem for et avansert, talestyrt heisssystem i IFI-bygningen. Kravene for denne snakkende heisen er som følger:

- Heisen begynner med å spørre personen hvilken etasje personen ønsker å gå til ("*Hvilken etasje skal du til?*").
- Etter at personen svarer f.eks. "*4. etasje*" må heisen spørre om å få en eksplisitt bekreftelse ("*Skal vi til 4. etasje?*").
- Hvis personen bekrefter ("*ja*") kan heisen gå til etasjen, takke brukeren og avslutte samtalen ("*Takk og velkommen tilbake!*").
- Brukeren bør også kunne spørre heissystemet hvor administrasjonen eller språkteknologigruppen befinner seg (f.eks. "*I hvilken etasje er språkteknologigruppen?*"). Systemet bør deretter svare brukeren og be om bekreftelse på at man ønsker å gå dit ("*Språkteknologigruppen er i 7. etasje. Skal vi dit?*").
- Hvis brukeren ikke blir forstått på noe tidspunkt under dialogen, bør systemet be brukeren om å avklare forespørselen deres (for eksempel "*Beklager, jeg forsto ikke hva du sa. Kan du gjenta?*")

Dere skal lage dialogsystemet med en endelig tilstandsautomat hvor kantene representerer mønstre (skrevet som regulære uttrykk) å anvende på brukerytringer. For å gjøre arbeidet deres lettere skal dere bruke Python-koden i `https://github.uio.no/IN2110/in2110-lab/blob/master/ekstra/fsm.py` som tilbyr alle nødvendige funksjoner for å bygge automaten deres. Et enkelt eksempel er også tilgjengelig nederst i filen.

Oppgaver:

1. Bruk de to metodene `add_state` og `add_edge` til å bygge automaten deres.
2. Så snart automaten er bygd kan dere teste den interaktivt med metoden `start_dialogue`.
3. Når dere er fornøyd med designet deres kan dere kjøre metoden `to_smcat` for å lagre automaten i et lesbart format, kopier resultatet og gå til nettsiden `https://state-machine-cat.js.org/` for å lage en visuell gjengivelse av automaten deres (dere kan endre diagrammets retning fra "top-down" til "left-to-right" i menyen til venstre). Til slutt kan dere laste ned diagrammet i en PNG-fil.

Svaret på dette spørsmålet må inneholde (a) PNG-filen som gjengir automaten deres sammen med (b) en kort forklaring (maksimalt en halv-side) av designvalgene deres i utviklingen av automaten. Du trenger ikke å levere noen kode.