IN1140 H2021 – Oblig 2a
Språkmodeller og Ordklassetagging

Innleveringsfrist, 20.10 kl. 23.59

Lever inn svarene dine i Devilry (`https://devilry.ifi.uio.no/`) i en fil som angir brukernavnet ditt, slik: `oblig2a_brukernavn.py`.
En perfekt besvarelse av denne oppgaven er verdt 100 poeng.

1 Språkmodeller (Eksamen V2017 – 10 poeng)

I Tabell 1 finner du formelen for en språkmodell (en såkalt bigrammodell) og et lite tekstkorpus. I denne oppgaven skal du ta utgangspunkt i denne formelen og i tekstkorspuset for å besvare følgende spørsmål:

Formel:

$$P(w_1 \dots w_k) = \prod_{i=1}^{k} P(w_i|w_{i-1})$$

Tekskorpus:

```
<s> Per synger ikke <\s>
<s> Kari synger <\s>
<s> Ola synger ikke <\s>
```

Table 1: Formel og tekstkorpus.

1. Hvilke bigrammer forekommer i korpuset?
2. Hvordan beregner vi sannsynligheten for et ord gitt det foregående ordet ($P(wi|w_{i-1})$) fra et korpus?
3. Du skal nå bruke bigrammodellen og tekstkorpuset til å beregne sannsynligheten for setningen "<s> Kari synger ikke <\s>". Vis hvilke sannsynligheter du trenger og hvordan disse beregnes fra korpuset. Du trenger ikke å regne ut den totale sannsynligheten for setningen.

2 Ordklasser (eksamen H2017) – (10 poeng)

Her skal vi jobbe med følgende setning:

Jeg spiser sushi med pinner

Gitt ordklassene i Tabell 2 under, tildel ordklasser til alle ordene i setningen. Du må velge ett alternativ for hvert ord.

CC	konjunksjon
DET	determinativ
JJ	adjektiv
NN	substantiv
PR	preposisjon
PO	pronomen
RB	adverb
SB	subjunksjon
VB	verb

Table 2: Ordklasser

3 Språkmodeller med Python (50 poeng)

I denne oppgaven skal du trene en **bigrammodell** med NLTK. Ta utgangspunkt i stegene og Python-koden som ble forklart i gruppeoppgave 4. I motsetning til i gruppeoppgaven skal vi her trene en bigrammodell og ikke en trigrammodell, så du blir nødt til å gjøre noen endringer i koden. Vi skal videre bruke Gutenberg-korpuset, og fokusere på dokumentet "bible-kjv.txt" som inneholder bibeltekst. Du skal i denne oppgaven skrive Python-kode som gir deg svar på følgende spørsmål:

1. Hva er totalt antall tokens i teksten?
2. Hva er totalt antall ord-typer (types) i teksten?
3. Hva er de 20 mest frekvente ordtypene i teksten?
4. Hva er frekvensen for henholdsvis ordene "heaven", "death", og "life"?
5. Hvilke bigram forekommer i den fjerde setningen?
6. Hvilke trigram forekommer i den femte setningen?
7. Tren en bigrammodell og generér en tekst ved bruk av denne.
8. Hva er sannsynligheten til den genererte teksten?

4 Ordklassetagging med regulære uttrykk (30 poeng)

I denne oppgaven skal du lage en ordklassetagger ved bruk av regulære uttrykk samt evaluere taggeren din på et ordklassetagget korpus.

Taggeren med regulære uttrykk i NLTK-boka (del 5.4.2, under overskriften *The Regular Expression Tagger*) sjekker kun noen få uttrykk, så her er det masse rom for forbedring: for eksempel kan vi tagge alle ord på *-able* som adjektiv.

I del 5.4.2 av NLTK-boka er de regulære uttrykkene definert i en liste av "regler" ellet "mønstre" slik:

```
>>> patterns = [
...     (r'.*ing$', 'VBG'),               # gerunds
...     (r'.*ed$', 'VBD'),                # simple past
...     (r'.*es$', 'VBZ'),                # 3rd singular present
...     (r'.*ould$', 'MD'),               # modals
...     (r'.*\'s$', 'NN$'),               # possessive nouns
...     (r'.*s$', 'NNS'),                 # plural nouns
...     (r'^-?[0-9]+(.[0-9]+)?$', 'CD'),  # cardinal numbers
...     (r'.*', 'NN')                     # nouns (default)
... ]

>>> regexp_tagger = nltk.RegexpTagger(patterns)
```

Taggeren kan testes på et korpus slik:

```
>>> regexp_tagger.tag(CORPUS_NON_TAGGED_SENTENCE)
```

For å kunne teste nøyaktigheten til taggeren din kan du bruke `evaluate`-metoden slik:

```
>>> regexp_tagger.evaluate(CORPUS_TAGGED_SENTENCES)
```

1. Definer en tagger ved hjelp av `nltk.RegexpTagger` der du har minst 10 uttrykk i tillegg til de som er nevnt i boka. Dokumentér alle reglene dine, og gi minst ett eksempel (for hvert uttrykk) på ord som dekkes.
   Husk å håndtere liten og stor bokstav. Enkeltord kan også brukes i de regulære uttrykkene, slik at f.eks. *the* alltid vil tagges som determinativ.

2. Bruk `adventure`-kategorien i Brown mens du utvikler taggeren din ved å teste nøyaktigheten til taggeren (med `evaluate`-metoden). Når du er fornøyd med reglene dine skal du teste taggeren på kategorien `fiction` i Brown, og rapportere resultatene. Det er viktig at du ikke endrer reglene dine etter dette.
   Hint: Ta utgangspunkt i NLTK-boka (kapittel 5, del 5.4.2).
   Merk: Poengene du får på denne oppgaven er ikke basert på nøyaktigheten til taggeren i den endelige evalueringen på `fiction`-kategorien. Det er helt vanlig at nøyaktigheten til en tagger synker når man tester den på et annet korpus enn den det ble utviklet på.

3. Til slutt skal programmet ditt lese inn filen `setninger.txt` som er lagt ut sammen med denne obligen, tagge den, og skrive ut resultatet. Kopier resultatet inn i filen din og diskuter minst 3 av feilene taggeren gjør, og kom med forslag til hvordan den kan forbedres.