Løsningsforslag til innlevering 1a, IN2110 V20

Oppgave 1 Data og preprosessering

a) Data

Se kode.

b) Pre-prosessering

Metode	Tokens	Typer
split	1829550	186519
word_tokenize	2086943	127814
word_tokenize+downcase	2086943	115793

Tabell 1: Antall tokens og typer for med forskjellige tokeniseringsmetoder.

I tabell 1 ser vi antall tokens og typer for forskjellige typer tokenisering på treningssettet. Vi ser at ved å bruke `word_tokenize` får vi en reduksjon i antall typer sammenlignet med å splitte på mellomrom, men at antall tokens går opp. Ved å gjøre om til små bokstaver reduseres antall typer ytterligere. Videre i oppgaven bruker vi den sistnevnte tokeniseringsmetoden.

c) Statistikk

Kategori	frekvens
games	1413
literature	2821
restaurants	428

Tabell 2: Fordeling mellom kategorier i treningssettet.

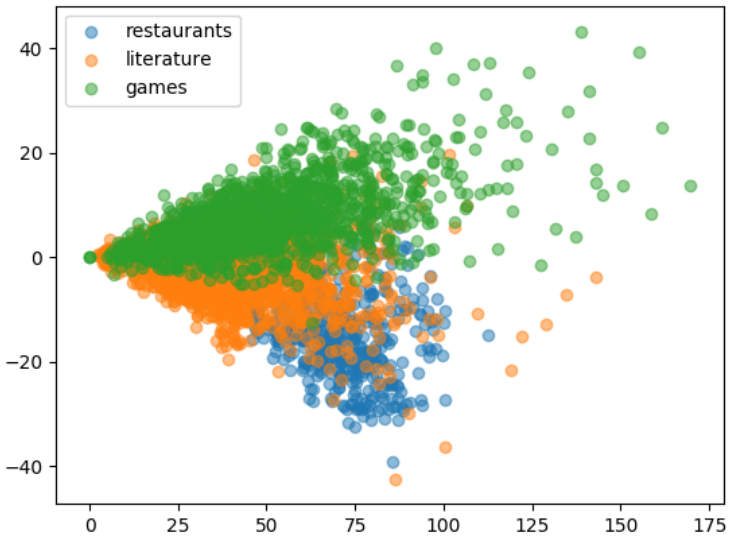
I tabell 2 ser vi fordelingen av anmeldelsene fra treningssettet over de tre kategoriene. Det er ganske stor forskjell mellom dem, med veldig mange fler i kategorien "literature" enn i "restaurants". Dette kan by på utfordringer i forhold til klassifisering.

Oppgave 2 Dokumentrepresentasjon

a) Vektorisering

Se kode.

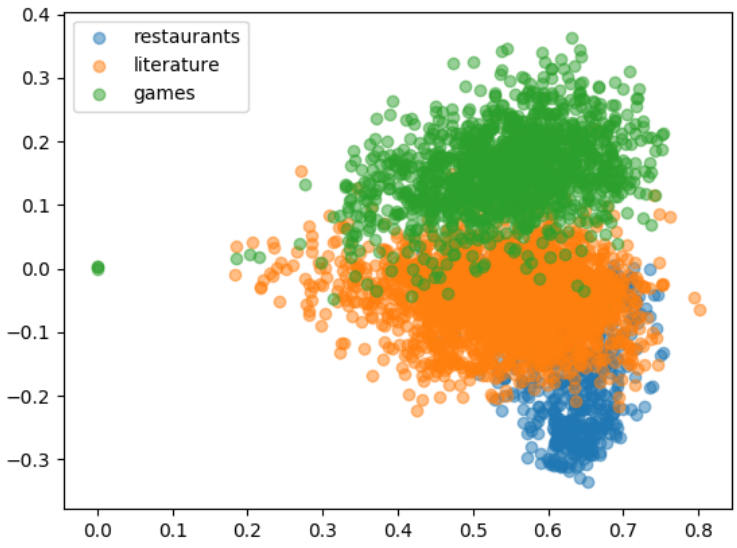
b) Visualisering



Figur 1: Visualisering av dokumentvektorene i treningssettet.

Figur 1 viser en visualisering av dokumentvektorene fra treningssettet, med forskjellig farge på punktene avhengig av hvilken kategori dokumentet tilhører. Klassene danner ikke tydelige klynger, men glir litt over i hverandre. I tillegg er de sensitive i forhold til dokumentlengde, noe som kan ses utifra kjegleformen på plottet.

c) Vekting



Figur 2: TF-IDF-vektede dokumentvektorer i treningssettet.

Figur 2 viser en visualisering av dokumentvektorene fra treningssettet etter å ha vektet dem med TF-IDF. Nå er det mye tydeligere klynger for hver kategori, og vi kan anta at de vil være lettere å skille for en klassifikator.

Oppgave 3 Klassifisering

Se kode.

a) Evaluering

	TF-IDF	
k	uten	med
1	0.880	0.869
2	0.911	0.814
3	0.880	0.974
4	0.900	0.979
5	0.888	0.983
6	0.893	0.979
7	0.883	0.983
8	0.890	0.983
9	0.885	0.983
10	0.888	0.985
11	0.880	0.985
12	0.887	0.988
13	0.881	0.986
14	0.890	0.988
15	0.881	0.988

Tabell 3: Accuracy med forskjellige verdier for k med og uten TF-IDF.

I tabell 3 ser vi resultatet av kNN-klassifikasjon med forskjellige verdier for k, både med og uten TF-IDF. Vi ser at det er kjøringene med TF-IDF som gir den beste ytelsen. Med vekting ser vi videre at ytelsen stort sett stiger jevnt med høyere verdi for k, opp til $k = 12$ da accuracy flater ut og vi ikke ser ytterligere forbedringer. For kjøringene uten vekting derimot ser vi at vi får best resultater ved lavere verdier for k (og ved k satt til 1 og 2 får vi til og med bedre resultater enn med vekting).

b) Testing

Ved å bruke $k = 12$ med TF-IDF får vi en accuarcy på 0.988 på test-settet. Dette er like bra som på dev-settet og er en indikasjon på at modellen generaliserer bra.