Rapport

Obligatorisk innlevering 1

Justyna, Adam, Joanna

Oppgave 1.

a)

Denne oppgaven har vi begynt med å skrive prepare_data() metode. Her har vi lagt to lister. Den første heter data og den andre heter labels. Listen med navn Data inneholder dokument-tekster og labels inneholder kategorier. Vi har brukt en enkel for løkke for å gå gjennom alle dokumenter. For å få ut de riktige dokumentene skrevet vi en if test som sjekker om dokumenter er en del av en av de tre bestemte kategorier: Games, Restaurants og Literatur. Dokument-tekster blir lagret i data listen og kategorier i labels listen. Til slutt metoden returner begge lister.

b)

I denne delen tokeniserte vi test-settet ved å bruke 4 forskjellige tokeniseringsmetoder, nemlig split(), word_tokenize(), lower(), model_tokenize()

Vi har brukt det første dokumentet istendenfor hele test settet fordi tokenize metoden tar tekst streng som en argument.

Vi har fått slike resutater:

split() = Tokens(483), Types (292)

word_tokenize() = Tokens(548), Types (285)

lower() = Tokens(548), Types (275)

model_tokenize() = Tokens(548), Types (285)

Lower() gir den laveste antall ordtyper. Denne metoden gjør om alle ord til små bokstaver.

c)

I denne delen har vi beregnet antall dokumenter per kategori. Her er resultater:

games = 1413 (30.3%)

literature = 2821 (60.5%)

Restaurants = 428 (9.2%)

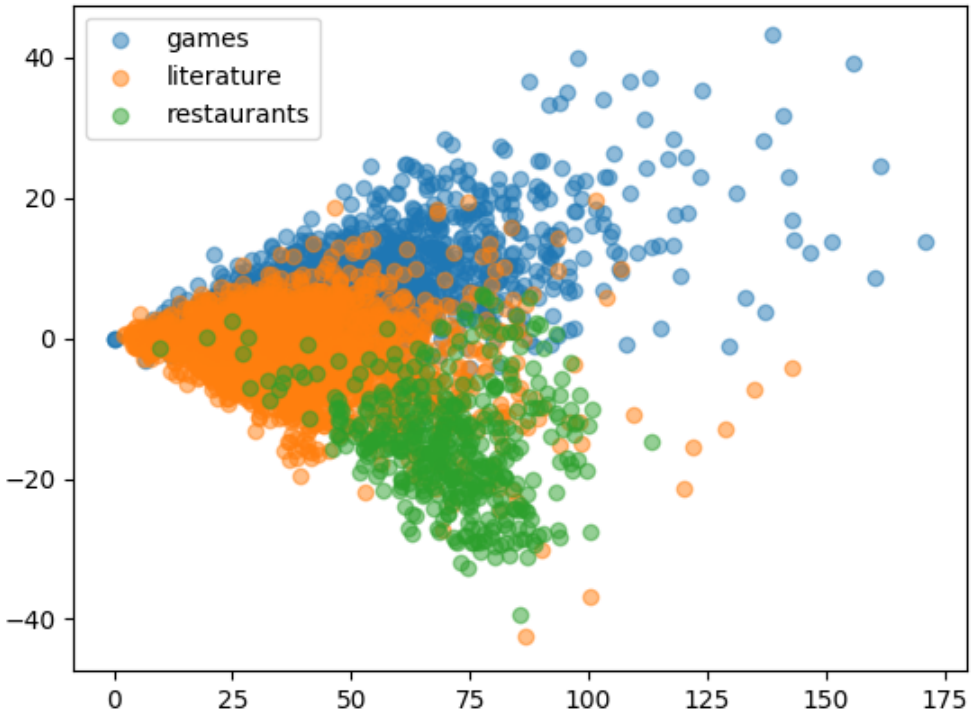
Oppgave 2.

a)

Her har vi laget CountVectorizer() i konstruktøren og i train() metoden trener vi den opp med fit_transform.

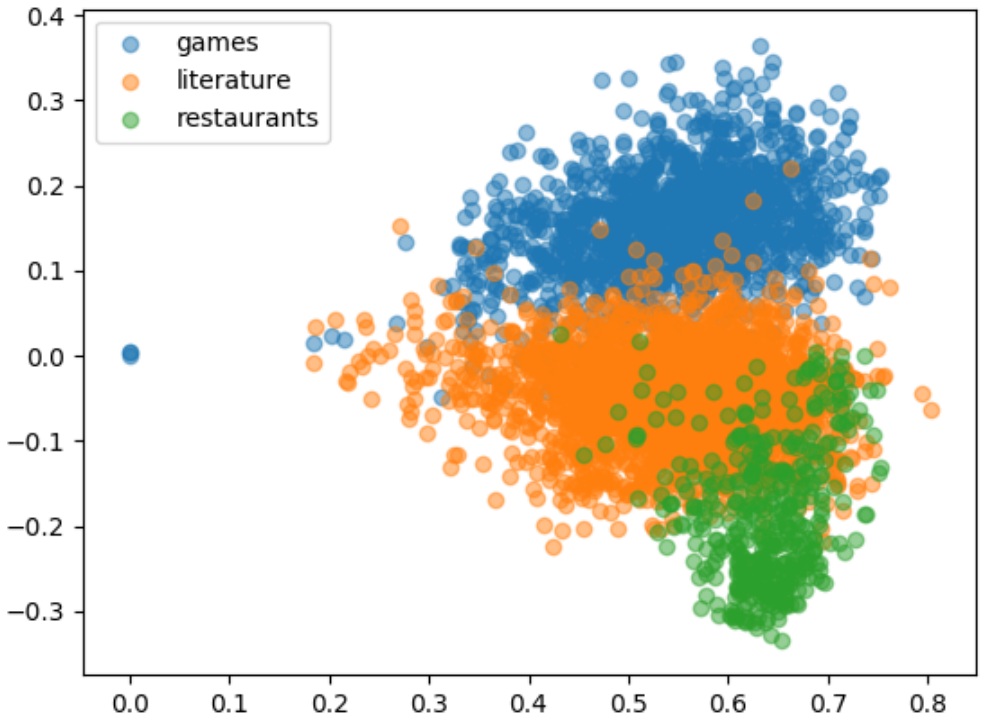
I vectorize() metoden har vi anvendt vektoriseren med transform() metoden.

b)



c)

I dette steget legget vi til en vekting metode som bruker TfidfTransformer og trener den opp på samme måte som CountVectorizer(). Forsjkellen mellom vektorisering med og uten Tfidf er at klasser er mer synlig på grafen ved å bruke vekting.



Oppgave 3.

a)

I denne oppgaven har vi lagd knn classifier med KNeighborsClassifier og har trent den opp med fit metoden

b)

Her har vi trent klassifikatoren med forskjellige verdier. Tabellen viser resultater:

k - verdi	med/uten tf-idf vekting	accuracy
1	Uten	0,88
1	Med	0,87
3	Uten	0,88
3	Med	0,97
10	Uten	0,89
10	Med	0,99
20	Uten	0,88
20	Med	0,98
30	Uten	0,86
30	Med	0,97
50	Uten	0,85
50	Med	0,96
100	Uten	0,81
100	Med	0,97
1000	Uten	0,7
1000	Med	0,86
4662	Uten	0,61
4662	Med	0,61

k = 10 med TfIdf vekting gir best accuracy. Noe som pekte seg ut var at k = 30 og k = 100 med vekting ga samme resultater. Ellers synker accuracy ned jo større k er og accuracy med vekting gir bedre resultater.

c)

Klassifikatoren ser ut til å fungere bedre på valideringsettet enn på testsettet, accuracy gir 0,81 på test data, mens på valideringsdata er resultatet 0,99.