Samuli Schroderus – Joonas Lehikoinen – Pasi Saarela – Timo Harju

Analyysiraportti (CRISP-DM)

Tehtävä n:o 10

Metropolia Ammattikorkeakoulu
Tietotekniikan koulutusohjelma
Ohjelmistotekniikka
Tiedonlouhinta
25.2.2020

1 Tavoitteet

Tekstinlouhinta harjoituksen alussa valittiin Project Gutenbergin e-kirjakokoelmasta satunnaisesti kahden eri kirjailijan kolme englanninkielistä teosta. Valitut kirjailijat olivat Oscar Wilden ja Mark Twain ja heiltä valittiin teokset: *. Tavoitteena on selvittää voidaanko tuntematon kirjailija selvittää klusteroinniin avulla esiprosessoidusta tekstiaineistosta, sekä todentaa, minkälaisia klustereita sanavektoreista syntyy.

* *"Adventures of Huckleberry Finn by Mark Twain", "Life on the Mississippi by Mark Twain", "The Adventures of Tom Sawyer by Mark Twain", "The Happy Prince, and Other Tales by Oscar Wilde", "The Importance of Being Earnest A Trivial Comedy for Serious People by Oscar Wilde" ja "The Picture of Dorian Gray by Oscar Wilde".*

2 Data

Gutenberg sivustolta kirjaa tekstimuodossa, joista kolme oli Mark Twainin kirjoittamia ja kolme Oscar Wilden kirjoittamia. Kirjojen lopussa oli gutenbergin lisenssi teksti.

Alla olevasta taulukosta on nähtävissä kirjailijoiden ja kirjojen nimet, sekä kirjojen tekstitiedostojen sisältämät merkkimäärät ja viittaukset.

Mark Twain	**Oscar Wilde**
Adventures of Huckleberry Finn Viittaus = Mark1 *Merkit = 566 407*	The Happy Prince, and Other Tales Viittaus = Oscar1 *Merkit = 86 728*
Life on the Mississippi Viittaus = Mark2 *Merkit = 795 221*	The Importance of Being Earnest A Trivial Comedy for Serious People Viittaus = Oscar2 *Merkit = 118 517*
The Adventures of Tom Sawyer Viittaus = Mark3 *Merkit = 386 670*	The Picture of Dorian Gray Viittaus = Oscar3 *Merkit = 450 301*

Taulukko 1. Kirjailijat, teosten nimet, merkkien määrät ja viittaukset.

Kirjoihin viitataan myöhemmin taulukossa esiintyvillä viittauksilla.

3 Datan valmistelu

Data editointiin käyttämällä Pythonin työkaluja ja NLTK-kirjastoa.

Vaihe 1:

```
# Avaa kirja
text = open(kirjalista[i], encoding="utf-8").read()

# Kirjan data pienaakkosiin
text=text.lower()

# Pilko data sanoihin
tk = WordPunctTokenizer()
text_tokenized=tk.tokenize(text)

# Porter Stemmer
porter = nltk.PorterStemmer()#PorterStemmer()
stemd_data =[porter.stem(t) for t in text_tokenized]
```

Kuva: vaihe 1

Kirjojen tekstitiedostot avattiin pythonissa open komennolla, käyttäen utf-8 formaattia. Tekstien kirjaimet muutettiin pienaakkosiksi ja pilkottiin sanalistoiksi NLTK:n Tokenizer työkalulla. Tämän jälkeen sanalistat normalisoitiin NLTK:n Porter-stemmerillä, joka poisti sanoista yleisimmät morfologiset ja taipuisat päätteet.

Vaihe 2:

```
# Tee lista kirjan tekstissä esiintyvistä sanoista
countsT=Counter()
countsT.update(stemd_data)
words=sorted(countsT,key=countsT.get,reverse=True)#reverse=True

# Filteröi kaikki paitsi aakkosmuotoiset sanat
words = [item for item in words if item.isalpha()]#Poista numerot
```

Kuva: vaihe 2

Seuraavaksi jokaista kirjaa kohti luotiin lista siinä esiintyvistä sanoista ja niiden esiintymis kerroista tekstissä käyttäen Pythonin Counter työkalua. Tämän jälkeen Countterin luomista listoista poimittiin niiden sisältämät sanat ilman esiintymis kertoja erillisille sana-listoille sorted() toiminnon avulla.

Lopuksi jokaisen kirjan sanalistoista suodatettiin kaikki muut merkit paitsi aakkoset isalpha() toiminnolla.

Vaihe 3:

```
m = Word2Vec(books_list,size=1682,min_count=2,sg=1)
#size = vektorin pituus(vähiten sanoja sisältäneen listan pituus)
#sg=1 ignore kielioppi

#Hae numero muotoinen vektori data Wor2Vector listasta
def vectorisoi(sent,m):
    vec=[]
    numw = 0
    for w in sent:
        try:
            if numw ==0:
                vec=m[w]
            else:
                vec = np.add(vec, m[w])
            numw+=1
        except:
            pass

    #Normalisoi vektorin data jakamalla vektori sanojen määrä / 2000
    return np.asarray(vec)/(numw/2000)

# Luo opetus data
l=[]
for i in stemlist:
    l.append(vectorisoi(i,m))
X=np.array(l)

```

kuva: vaihe 3

Kerätty data prosessoitiin GenSim:n Word2Vector työkalulla ja luotiin opetus data. arvot normalisoitiin jakamalla lukuarvot kirjan sanalistan sanamäärällä, joka oli jaettu 2000:lla. ylimääräinen 2000 jakaminen ei ole välttämätöntä. Se suoritettiin, koska muuten vektorien arvoista tulisi todella pieniä ja niitä olisi hankala havainnollistaa mallinnus vaiheen kuvaajassa.

4 Mallinnus

Klusterointi tehtiin Pythonin Scikit-Learn kirjastolla.

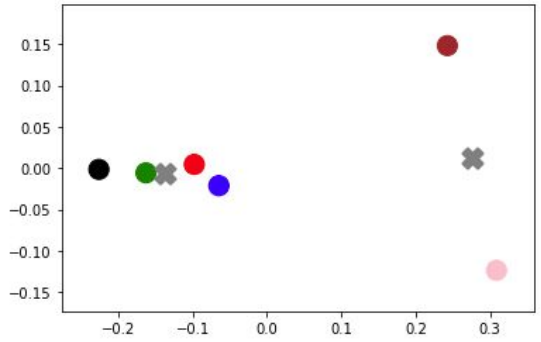
Klusteroinnissa käytettiin K-means klusterointi algoritmia. Klusterien määräksi asetettiin 2, MaxIteraatioksi 5000, klusterien aloituspisteen valitsemis parametriksi(init) valittiin k-means++ ja aloitus pisteiden valitsemiskerroiksi (n_init=1).(kuva: mallinnus)

```
#Klusterien määrä
n_clusters = 2

#Kmeans
clf = KMeans(n_clusters=n_clusters,max_iter=5000,init="k-means++",n_init=1)

#Klusteroi data
labels=clf.fit_predict(X)
```

Kuva Mallinnus 1. Mallin parametrit

Tuloksista havaittiin, että Oscar3 keskittymä eli Oscar Wildin teos "The Picture of Dorian Gray" oli sijoittunut väärin "Mark Twainin" kirjat sisältämään klusteriin. (Kuva mallinnus taulukko 1.)

```
blue  =  Mark1
black  =  Mark2
green  =  Mark3
pink  =  Oscar1
brown  =  Oscar2
red  =  Oscar3
```



Kuva mallinnus 2. Klusteroinnin lopputulos visuaalisessa muodossa (X = klusterin keskipiste). Nimien viitteet löytyvät alla olevasta taulukosta (Mallinnus Taulukko 1.)

Muuttuja	Teos	Klusteri
Mark1	*Adventures of Huckleberry Finn*	0
Mark2	*Life on the Mississippi*	1
Mark3	*The Adventures of Tom Sawyer*	1
Oscar1	*Adventures of Huckleberry Finn by Mark Twain*	0
Oscar2	*The Happy Prince, and Other Tales*	0
Oscar3	*The Picture of Dorian Gray*	1

Mallinnus Taulukko 1. Muuttujien selvennys sekä muuttujien klusterisijainti.

5 Arviointi

Saaduista klusterointi tuloksista havaitaa, että tekstiaineiston klusterointi toimi melko hyvin, vaikka yksi Oscar Wild:lle klusteriin kuuluva keskittymä oli sijoittunut Mark Twainin Clusteriin. Tuloksiin on saattanut vaikuttaa, että Oscar Wildin teos "The Picture of Dorian Gray" on sattumalta kirjoitystyyliltään samankaltainen muiden Mark Twainin teosten kanssa. Isommilla kyseisten kirjailijoiden teosten aineistoilla oltaisiin mahdollisesti saatu tarkempia tuloksia.

6 Täytäntöönpano

Klusterointi soveltuu jossain määrin selvittämään kirjailijan tekstiaineistosta. Tässä harjoituksessa aineiston koko oli kuitenkin verrattavan pieni ja sattumalla valitut kirjailijoiden kirjat saattoivat olla lopulta myös melko samankaltaisia, sillä tekstiä ei täysin oikoluettu. Isomman aineiston tiedonlouhinta projekteissa aineistosta oltaisiin saatu vähennetty sattuman vaikutusta ja näin saatu kattavampia ja tarkempia sanavektoreita. Tuloksina saatiin silti kohtuullisen hyviä tuloksia, missä vain yksi

keskittymä oli sijoittunut väärään klusteriin ja muut keskittymät olivat sijoittunut oikeisiin klustereihin.

Tehtävän JupyterNoteBook