Diplomski studij

Informacijska i komunikacijska tehnologija

Telekomunikacije i informatika

Računarstvo

Programsko inženjerstvo i informacijski sustavi

Računalno inženjerstvo
Računarska znanost

Ak. g. 2017./2018.

Raspodijeljena obrada velike količine podataka

3. Domaća zadaća

1. Zadatak: Indeksiranje tekstualne kolekcije

Cilj zadatka je uspješno napisati, prevesti te izvršiti Javin program za izračun matrice sličnosti objekata preporuke. Sadržaj objekata je tekstualnog tipa pa je zadatak potrebno obaviti indeksiranjem i pretraživanjem tekstualne kolekcije. Izračunata matrica sličnosti će se koristiti u 2. zadatku ove domaće zadaće za računanje preporuka po sadržaju.

Uspješnim rješenjem ovog zadatka steći ćete sljedeća znanja:

- **osnove rada s programskim okvirom Apache Lucene**
- **izrada matrice sličnosti objekata preporučivanja na osnovu njihova (tekstualnog) sadržaja**

IZVJEŠTAJ: Na sustav Moodle predajete izvorni programski kôd te odgovore na pitanja koja slijede iza opisa zadataka.

Za potrebe ove zadaće će biti potrebno dohvatiti arhivu s ulaznim podacima sa sljedeće poveznice: http://eigentaste.berkeley.edu/dataset/jester_dataset_2.zip. Opis podataka u ovoj arhivi se nalazi na poveznici http://eigentaste.berkeley.edu/dataset/ pod imenom Dataset 2. U ovom zadatku je potrebno koristiti datoteku `jester_items.dat` iz ove arhive.

Detaljni opis zadatka:

U ovom zadatku ćete napraviti Javin program u obliku Maven projekta koji će ulaznu datoteku `jester_items.dat` koja se sastoji od niza šala na engleskom jeziku i njihovih ID-jeva obraditi na sljedeći način:

1. parsirati ulaznu tekstualnu datoteku sa šalama,
2. stvoriti i indeksirati Luceneove dokumente od parsiranih šala,
3. izračunati sličnosti između Luceneovih dokumenata i pohraniti ih u matricu sličnosti,
4. normalizirati matricu sličnosti i
5. pohraniti sličnosti Luceneovih dokumenata (tj. šala) u izlaznu datoteku.

U prvom dijelu zadatka treba parsirati ulaznu tekstualnu datoteku i napuniti mapu `Map<Integer,String>` podacima. Ključ mape treba biti ID, a vrijednost treba biti tekst šale. Primijetite da je tekst šala zapisan u HTML obliku. Ispravan *escape*-ing iz ovog oblika u čisti tekst ćete ostvariti korištenjem klase `StringEscapeUtils` iz Apacheovog paketa `commons-lang3`:

```
StringEscapeUtils.unescapeXml(jokeText .toLowerCase().replaceAll("\\<.*?\\>", ""))
```

Ovaj paket uključite u vaš Maven projekt kao *dependency*. U nastavku će vam također trebati Luceneovi paketi `lucene-core`, `lucene-analyzers-common` i `lucene-queryparser` pa i njih uključite u vaš Maven projekt kao *dependency*.

Predložak za drugi i treći dio zadatka se nalazi na sljedećoj poveznici: http://www.lucenetutorial.com/lucene-in-5-minutes.html. U drugom dijelu zadatka je potrebno od objekata u mapi stvoriti Luceneove dokumente (klasa `org.apache.lucene.document.Document`). Svaki Luceneov dokument se sastoji od jednog ili više indeksabilnih polja (sučelje `org.apache.lucene.index.IndexableField`). Vaši dokumenti će imati dva polja `ID` i `text`. Prvo polje treba biti pohranjeno u indeks, ali ne treba biti indeksirano niti tokenizirano jer ga nećete koristiti za izračun sličnosti dokumenata. To ćete ostvariti na sljedeći način:

```
FieldType idFieldType = new FieldType();
idFieldType.setStored(true);
idFieldType.setTokenized(false);
```

```
idFieldType.setIndexOptions(IndexOptions.NONE);
Field textfield = new Field("ID", entry.getKey().toString(), idFieldType));
```

Drugo polje ne treba biti pohranjeno u indeks, ali treba biti indeksirano i tokenizirano jer ćete njega koristiti za izračun sličnosti.

U trećem dijelu zadatka trebate stvoriti matricu sličnosti tipa float[][] čije dimenzije odgovaraju veličini mape. Upit definirajte na osnovu teksta dokumenta na sljedeći način:

```
Query query = new QueryParser("text", analyzer).parse(QueryParser.escape(docText));
```

Prilikom dohvaćanja najsličnijih objekta (klasa `org.apache.lucene.search.TopDocs`), dohvatite njihov najveći mogući broj tako što ćete u metodu `search` klase `org.apache.lucene.search.IndexSearcher` kao drugi argument predati duljinu mape. Nakon toga prođite kroz sve najsličnije objekte i za svaki na kojeg naiđete dodajte odgovarajući zapis u matricu. Do pohranjenog ID-a dokumenta možete doći pozivom metode `document` klase `org.apache.lucene.index.IndexReader`. Kao argument ove metode predajte atribut `doc` klase `org.apache.lucene.search.ScoreDoc`.

U četvrtom dijelu je potrebno normalizirati matricu sličnosti da joj vrijednosti budu u rasponu 0 do 1 i da sličnosti budu simetrične (tj. da svaki par dokumenata ima istu sličnost prvog s drugim dokumentom i obratno). Tijekom normalizacije prvo podijelite svaki redak s maksimalnom vrijednošću da vrijednosti u matrici dovedete u očekivani raspon. Pri tome uočite da je maksimalna vrijednost retka sličnost odgovarajućeg dokumenta s njim samim. Nakon toga izračunajte sličnosti `i`-tog i `j`-tog dokumenta kao prosječnu vrijednost ćelije `ij` i `ji`.

U petom dijelu zadatka pohranite sličnosti iz ove matrice u tekstualnu datoteku imena `item_similarity.csv` sa zapisima u formatu: `ID1, ID2, sličnost`. Pri tome nemojte zapisivati sličnosti koje su jednake 0 jer je to suvišno, a niti duple vrijednosti jer su sličnosti para dokumenata identične zbog simetričnosti matrice sličnosti.

Nakon uspješnog pokretanja programa odgovorite na sljedeća pitanja:

- Koliko se zapisa nalazi u izlaznoj datoteci?
- Koja šala je najsličnija šali s ID-jem 1?
- Vidite li zašto su te dvije šale slične?
- Što mislite, hoće li preporuka po sadržaju imati smisla u slučaju ovih šala?

2. Zadatak: Izgradnja i evaluacija centraliziranog preporučitelja

Cilj zadatka je napisati i evaluirati 2 centralizirana preporučitelja od kojih je jedan temeljen na suradnji korisnika, a drugi na sličnosti objekata (*item-based recommendation*).

Uspješnim rješenjem ovog zadatka steći ćete sljedeća znanja:

- **osnove rada s programskim okvirom Apache Mahout**
- **izrada i evaluacija jednostavnog preporučitelja**

IZVJEŠTAJ: Na sustav Moodle predajete izvorni programski kôd te odgovore na pitanja koja slijede iza opisa zadataka.

Detaljni opis zadatka:

U ovom zadatku ćete napraviti 2 Javina programa u obliku Maven projekta. U ovaj projekt je potrebno uključiti Apacheov paket `mahout-core` kao *dependency*. Prvi program predstavlja preporučitelja temeljenog na sličnosti objekata, a drugi preporučitelja temeljenog na suradnji korisnika. Oba programa koriste ulaznu datoteku `jester_ratings.dat` s ocjenama korisnika, a prvi program koristi i datoteku `item_similarity.csv` iz prvog zadatka. U drugom programu definirajte sličnost korisnika koristeći mjeru Pearsonove korelacije, a slične korisnike definirajte kao one koji su od korisnika udaljeni manje od granice `0,1`. Nakon toga napišite programski kod za evaluaciju kvalitete ova dva preporučitelja (bez računanja odziva i preciznosti). Pri pozivu metode `evaluate` klase `org.apache.mahout.cf.taste.eval.RecommenderEvaluator` proizvoljno odaberite vrijednosti parametara `trainingPercentage` i `evaluationPercentage` u rasponu od `0,3` do `0,7`. Međutim, neka vrijednosti tih parametara budu iste za oba preporučitelja. U slučaju drugog preporučitelja također napišite programski kod koji će ispisati 10 preporuka prvih 100 korisnika na disk.

Nakon uspješnog pokretanja odgovorite na sljedeća pitanja:

- Kojih 10 preporuka je za korisnika s ID-jem 220 je izračunao prvi, a koje drugi preporučitelj?
- Koje preporučitelj ima bolju kvalitetu?
- Je li za ove ulazne podatke bolje koristiti mjeru log-likelihood ili Pearsonovu korelaciju u slučaju drugog preporučitelja?

3. Zadatak: Pokretanje raspodijeljenog preporučitelja

Cilj zadatka je pokrenuti raspodijeljenog preporučitelja u računalnom grozdu.

Uspješnim rješenjem ovog zadatka steći ćete sljedeća znanja:

- **Pokretanje raspodijeljenog preporučitelja**

IZVJEŠTAJ: Na sustav Moodle predajete unose u komandnu liniju te odgovore na pitanja koja slijede iza opisa zadataka.

Detaljni opis zadatka:

Pokrenite raspodijeljenog preporučitelja u pseudo-raspodijeljenom načinu rada na vašem računalu. Neka ovaj preporučitelj izračuna 10 preporuka za prvih 100 korisnika. Korisnike definirajte koristeći parametar `--usersFile` prilikom pokretanja posla iz komandne linije. Mjera sličnosti neka odgovara preporučitelju temeljenom na suradnji korisnika iz prethodnog zadatka.

Nakon uspješnog pokretanja odgovorite na sljedeće pitanje:

- Koliko ste MapReduce poslova izvršili u vašem kôdu?
- Pogledajte izlazne datoteke i objasnite postoji li razlika u izračunatim preporukama u odnosu na preporučitelja iz 2. zadatka?