Data Science and Big Data

Summer Term 2020

1 – Jaccard Distance with Spark
Michael Mock, 3.6.2020

Hand in: June 9th, 2020 (17:00 CET)
Will be discussed: June, 10th, 2020

General procedure

Java 8 JDK and Eclipse (at least Kepler) are required. Download the Maven SparkExamples Project from the lecture website, `unzip SparkTemplate.zip`, import the maven project (pom.xml) into Eclipse and execute "Maven Update", on the Bda project. The template provides running examples and code fragments for the exercises. Solutions must be provided as runnable maven projects (zipped) including pom.xml that can be imported into Eclipse in the same way. Please send also an additional e-mail (whithout any attachment) that indicates that you were sending in the solution, as sometimes mails with attachments are dropped.

Note: in order to setup the workspace, an Internet connection is required, as Maven is downloading the required open source packages automatically in the background. Execute Maven->update project to initiate the download.

Data

The exercises work on real-world data provided by http://last.fm. The material is copyrighted by last.fm and free to use for scientific and educational purposes. The datasets contains tab separated listening events, each line indicates that the named user has listened to the indicated artist at the indicated point of time. A small sample data is provided in the workspace. Exercises should work on the large sample provided separately in the file last-fm-sample1000000.tsv on the lecture website. More data can be downloaded from:
http://www.dtic.upf.edu/~ocelma/MusicRecommendationDataset/lastfm-1K.html
as distributed by O. Celma, O, Music Recommendation and Discovery in the Long Tail, Springer 2010.

When using the original data, some preprocessing is required:

```
# Remove " and ' quotes and |
cat userid-timestamp-artid-artname-traid-traname.tsv |sed 's/\"//g'|sed "s/'//g" | sed "s/|//g"
>clean.tsv
# sort 2nd column (timestamp)
cat clean.tsv | sort -k 2 > clean_sorted.tsv
```

You may experiment with the original data if you want. Solutions must be provided together with the small sample data, questions must be answered working with the large sample data last-fm-sample100000.tsv only.

Task 1) Horizontal Scalability (20P)

Explain briefly the notions of horizontal scalability, fault-tolerance, parallelization and location transparency in the context of Big Data Systems. How do fault-tolerance and horizontal scalability relate to each other?

Task 2) Implement a Spark Program "UserClicks" (15P)

- the program should compute the number of listening events per artist
  Input: last-fm-sample100000.tsv
  Output: artname, noOfListeningEvent
  one line per artname in the input data, stored as textfile
- Hint: start from the word count example
- how many listening events has the artist "Mark Knopfler" (hint: use grep "Mark Knopfler" on the output file)

Task 3) Implement a class UserSet that stores a userset and provides the following methods: (15P)

```
public Userset add (String username);
//add a user to the userset
public Userset add (Userset set);
//add a userset to the userset

Public double distanceTo(Userset other);
// compute Jaccard-Distance to another userset
```

Task 4) Implement a Spark Program "UserSet" (15P)

- the program should compute the userset for each artist, i.e.
- userset = names of users listening to an artist
  Input: last-fm-sample1000000.tsv
  Output: artname, [user_001, user_936, user_800, …]
  one line per artname in the input data, stored as textfile
  hint: use aggregateByKey or combineByKey
- what is the userset of "Mark Knopfler"? (hint: use grep "Mark Knopfler" on the output file)