Please upload the source code of your solutions on or before the due date to the provided Moodle assignment as a single zip file using your group id as the file name. Provide some brief instructions on how to run your solution to each problem in a file called `Problem_X.txt`*, and report also the most important results (such as number of results, runtimes, etc.) of your solutions to that problem within this file. Remember that all solutions may be submitted in groups of up to 3 students.*

***Note:* Since this exercise sheet imposes intensive computational workloads, it is highly recommended to run the following experiments on the IRIS HPC cluster by using either the interactive or batching modes of the Spark launcher scripts (see once more the "Getting Started" guide provided on Moodle for detailed instructions).**

Latent Semantic Analysis of Wikipedia Articles — 12 Points

Problem 1. Consider the Wikipedia dataset which is available from Moodle under the link `Wikipedia-En-41784-Articles.tar.gz` of Chapter 1. Also consider the sample script `RunLSA.scala` or `RunLSA.ipynb` notebook provided on Moodle as basis to solve this exercise.

(a) Translate the provided `RunLSA.scala` script into an actual Scala object and compile this object into a jar file called `RunLSA.jar`, which can also be executed via `spark-submit` on the IRIS cluster. Alternatively, extract the plain Python code from the `RunLSA.ipynb` notebook if you prefer Python by using `nbconvert --to script` and submit your Python script as a Spark job on the IRIS cluster.

In either case, create two versions of your code, one that performs an NLP pipeline for parsing the articles, and one that uses a simple tokenizer (using the regular expressions for words and numbers introduced in Exercise Sheet #1).

Next, compute the *top-25 terms* and the *top-25 documents*, each under the *top-25 latent concepts*, based on the examples provided in the notebook (and as shown in the lecture), and observe how the results vary depending on which tokenizer or lemmatizer you use. **4 Points**

(b) For the entire dataset consisting of all the 41,784 Wikipedia articles, run your Spark job with the following configurations and compare their runtimes:

- $numFreq \in \{5000, 10000, 20000\}$ for the number of frequent terms extracted from all Wikipedia articles, and
- $k \in \{25, 100, 250\}$ for the different numbers of latent dimensions used for the SVD.

Also compare among different runs with and without using the NLP pipeline. Manually inspect the results as in (a) to find the best hyper-parameters for your setting. **4 Points**

Note: The setting with $numFreq = 20000$ *and* $k = 250$ *should run in less than one hour for all of the 41,784 articles on the IRIS cluster.*

(c) After having found the best hyper-parameters for your setting, implement a basic search engine for your framework, which takes a list of keywords as input and retrieves the most relevant documents based on the best SVD you obtained in (b) (also refer to Slides 34–39 of Chapter 4 for an explanation of these steps). Report the results (including the titles of the Wikipedia articles) for 5–10 interesting keyword queries. **4 Points**

Summarize the results (incl. the Spark runtimes) of this experiment in your `Problem_1.txt` file.

Latent Semantic Analysis of Wikipedia Movie Plots **12 Points**

Problem 2. Consider the "*Wikipedia Movie Plots*" dataset as it is available from Kaggle:

`https://www.kaggle.com/datasets/jrobischon/wikipedia-movie-plots`

You may download the dataset (in CSV format) either from the above URL or via Moodle. This openly available dataset captures 8 fields (*release year*, *title*, *origin/ethnicity*, *director*, *cast*, *genre*, *wiki page*, and *plot*) scraped from 134,164 Wikipedia movie articles. For this problem, we will focus on the usage of LSA to analyze the *plot* fields of these articles. Follow the general methodology provided in the `RunLSA.scala` script or `RunLSA.ipynb` notebook to solve the following tasks.

(a) Adapt the provided parsing functions of the `RunLSA.scala` script or `RunLSA.ipynb` notebook from Moodle to read the *title*, *genre* and *plot* fields of the articles into an initial DataFrame in Spark (see Exercise Sheet #2 and/or the lecture slides for details). Also make sure to apply a proper schema to your DataFrame object. **4 Points**

(b) Next, add an additional column called `features` to your DataFrame, which contains a list of lemmatized text tokens extracted from each of the *plot* fields using the NLP-based `plainTextToLemmas` function of the provided script or notebook.

Also here, translate the `RunLSA.scala` script or `RunLSA.ipynb` notebook (including the new steps from (a)–(b)) into an actual Spark job which can then be executed via `spark-submit` on the IRIS cluster.

2 Points

(c) Compute an SVD decomposition of the 134,164 movie plots contained in your DataFrame by using the following two hyper-parameters:

- $numFreq = 5000$ for the number of frequent terms extracted from all Wikipedia articles, and
- $k = 25$ for the number of latent dimensions used for the SVD.

Based on the two `topTermsInTopConcepts` and `topDocsInTopConcepts` functions provided in the `RunLSA.scala` script and `RunLSA.ipynb` notebook, compute the *top-25 terms* and the *top-25 documents*, each under the *top-25 latent concepts*, for the above SVD.

Manually inspect the results to determine if the SVD improves the representation of the documents.

2 Points

(d) Extend the provided `topDocsInTopConcepts` function such that it prints also the top-5 original *genre* labels associated with each Wikipedia article returned by this function. **2 Points**

(e) Finally, think of 5–10 interesting keyword queries for movies and report their results (including the titles of the movies). **2 Points**

Summarize the results (incl. the Spark runtimes) of this experiment in your `Problem_2.txt` file.

Outlier Detection for Gearbox Readings **12 Points**

Problem 3. Consider the "*Gearbox Fault Detection*" dataset released by the NASA in 2009 and available from Kaggle:

`https://www.kaggle.com/datasets/hetarthchopra/gearbox-fault-detection-dataset-phm-2009-nasa`

You may download the dataset (in CSV format) either from the above URL or via Moodle. Apply and (if necessary) adapt the `RunKMeans.scala` script or `RunKMeans.ipynb` notebook to detect possible outliers among the 14M gearbox readings provided in this dataset. Start with a reasonably small amount of clusters (e.g. using $k = 2 \dots 12$), and also aim to provide a plot of a (reasonably small) sample of the readings by using a visualization technique of your choice (submitting the source code to generate the visualization from your data is sufficient; submitting the images is not required). For each value of k, also provide a list of the top-25 outliers among all clusters. **12 Points**

Summarize the results (incl. the Spark runtimes) of this experiment in your `Problem_3.txt` file.