Ton Duc Thang University
Faculty of Information Technology

ESSAY – PROGRESS II

Course: Mining Massive Datasets

Duration: 05 weeks

I. Formation

- The essay is conducted in groups of **04 – 05** students.
- The student group fulfills the requirements and submits the work according to the detailed instructions below.

II. Requirements

The **WebOfScience-5736** dataset contains 5736 corresponding documents, one per line.

Student groups implement the MinHashLSH algorithm in two versions including in-memory and large-data

a) Task 1 (3.0 points): In-memory MinHashLSH

In this version, student groups re allowed to use pure in-memory processing operations to implement the MinHashLSH algorithm and encapsulate it into the corresponding class.

The **InMemoryMinHashLSH** class is described as follows.

InMemoryMinHashLSH
+ InMemoryMinHashLSH (documents: DataFrame) + shingling(documents: DataFrame): DataFrame + minhashing(bool_vectors: DataFrame): DataFrame + locality_sensity_hashing(signatures: list): DataFrame + run(): void + approxNearestNeighbors(key, n): DataFrame

- Attributes: Student groups propose and implement necessary attributes for storing data within the object, aiming to limit redundant computation of results.
- Methods:

- InMemoryMinHashLSH(documents: DataFrame): constructor, taking a list of documents.
- shingling(documents: DataFrame): takes a list of documents, performs the Shingling step in the algorithm, and returns Boolean vectors.
- minhashing(bool_vectors: DataFrame): takes a list of Boolean vectors, performs the MinHashing step in the algorithm, and returns signatures.
- locality_sensity_hashing(signatures: DataFrame): takes a list of signatures, performs the corresponding step in the algorithm, and returns results in the structure <signature>-<bucket ID>.
- run(): executes the Min Hash LSH algorithm and stores the results in related attributes.
- approxNearestNeighbors(): takes the document to be queried (key) and the maximum number of results (n), returns n documents with the highest similarity in the document set.

- Students implement additional functions to measure Jaccard similarity/distance between documents to verify retrieval results.
- In this task, students are permitted to use the Pandas library and modify data structures and data types if necessary.
- Student groups develop a program illustrating the operation of the implemented class, visualizing and evaluating the results of the approxNearestNeighbors() function.

b) Task 2 (5.0 points): LargDataMinHashLSH

Student groups re-implement the requirements from task 1 for the case of large-data using the PySpark library. Note:

- Avoid using the Pandas library and similar purely in-memory libraries.
- Carefully make us of collect-like functions, ensuring that the program runs efficiently as the dataset size significantly increases.

c) Task 3 (2.0 points): Presentation

- Student groups compose a presentation to report your work.

- **THERE IS NO PRESENTATION TEMPLATES. STUDENTS ARANGE CONTENTS IN A LOGICAL LAYOUT BY YOURSELVES.**
- The presentation must include below contents
  - Student list: Student ID, Full name, Email, Assigned tasks, Complete percentage.
  - Briefly present approaches to solve tasks, should make use of pseudo code/diagrams.
  - AVOID EMBEDDING RAW SOURCE CODE IN THE PRESENTATION.
  - Study topics are introduced briefly with practical examples.
  - Advantages versus disadvantages
  - A table of complete percentages for each task.
  - References are presented in IEEE format.
- **Format requirements:** slide ratio of 4x3, avoid using dark background/colorful shapes because of projector quality, students ensure contents are clear enough when printing the presentation in grayscale.
- Presentation duration is **10 minutes**.

III. Submission Instructions

- Create a folder whose name is as

  QT2_<Group ID>

- Content:
  - **source.ipynb** → source code (remain all cell outputs)
  - **source.pdf** → pdf of the notebook
  - **presentation.pdf** → presentation.
- Compress the folder to a zip file and submit by the deadline.

IV. Policy

- **Student groups submitting late get 0.0 points for each member.**
- **Missing required materials in the submission loses at least 50% points of the presentation.**

- **Copying source code on the internet/other students, sharing your work with other groups, etc. cause 0.0 points for all related groups.**
- **If there exist any signs of illegal copying or sharing of the assignment, then extra interviews are conducted to verify student groups' work.**

-- THE END --