School of Computing and Information Technology
University of Wollongong
CSCI316 – Big Data Mining Techniques and Implementation – 2020

Assignment 2

10 Marks
Due: 13/Nov/2020 in Week 13

Three (3) tasks are included in this assignment. The specification of each task starts in a separate page.

You are supposed to complete this assignment in Jupyter Note. You must write and execute all your Python code in Jupyter Notebook and generate a PDF document of your work. ***The deliverables include one Jupyter Notebook source file (with .ipybn extension) and one generated PDF document for each task.***

Note: To generate a PDF file for a notebook source file, click "File" on top of the notebook, and choose "Download as" and then "PDF via LaTex".

All results of your implementation must be reproducible from your submitted Jupyter notebook source files. In addition, the PDF file that you generate must include all execution outputs as well as clear explanation of your implementation algorithms (e.g., in the Markdown format or as comments in your Python codes).

Submission must be done online by using the submission link on MOODLE. The size limit for all submitted materials is 20MB. Submissions made after the due time will be assessed as late submissions. Late submissions are counted in full day increments (i.e. 1 minute late counts as a 1 day late submission). There is a 25% penalty for each day after the due date including weekends. The submission site closes four days after the due date. No submission will be accepted after the submission site is closed.

This is an individual assignment. Plagiarism of any part of the assignment will result in having zero marks for the assignment and for all students involved.

Marking guidelines

Code: Your Python code will be assessed by the virtual machine environment installed on the lab computer assigned to you in Weeks 7&8's lab. It defines the standard environment for code development and code execution. Note that the correctness, completeness, efficiency, and results of your executed code will be assessed. Thus, code that produces no useful outputs will receive zero marks. This also means that code that does not run in the above virtual machine environment would be awarded zero marks or code where none of the core functions produce correct results would be awarded zero marks.

Your answers in the PDF document: The correctness and completeness of your answers will be assessed.

(Continue on next page)

Task1

(3 marks)

Dataset: Yoochoose Clicks Dataset (yoochoose-clicks.dat)
Source: http://recsys.yoochoose.net/challenge.html

About the Dataset Information
Download the data source from the above link. The size of the files after decompression is about 1.91GB. The data source contains three datasets. For this task, you just need the yoochoose-clicks.dat dataset.
This dataset is a set of click events collected from a website of an online retailer. Each record in the dataset has four (4) fields:

- Session ID - the id of the session. In one session there are one or many clicks.
- Timestamp - the time when the click occurred.
- Item ID – the unique identifier of item.
- Category – the category of the item.

The value "S" indicates a special offer, "0" indicates a missing value, a number between 1 to 12 indicates a real category identifier, and any other number indicates a brand. If an item has been clicked in the context of a promotion or special offer then the value is "S". If the category is a brand (i.e., BOSCH) then the value is an 8-10 digits number. If the item has been clicked under a regular category (e.g., sport) then the value is a number between 1 to 12.

The Task
You are to perform ***explorative analysis*** on the given dataset of click events by using Spark's DataFrame API. The objective is to compute *the average time that users stay on items in each category*.

For analysis purposes in this task, use the following definitions:
(i) There are 15 item categories in the dataset: S, 0, 1 to 12, and B (for any 8-10 digits number)
(ii) In each session, the time that a user stays on some item is the timestamp difference between a user clicking on this item and the next item (if there is a next item).

Requirements
(i) Load yoochoose-clicks.dat into a Spark DataFrame with correct types and *print its schema*.
(ii) Implement a sequence of DataFrame transformations plus one action that produces the final output (as specified above). ***Do not*** convert any DataFrame into Python data structure such as Pandas dataframe, NumPy array, list, collection, etc. **The entire analysis should be completed with the Spark DataFrame API.**

Deliverables
(1) A Jupiter Notebook source file named `<your_name>_task1.ipybn` which contains your implementation source code in Python;
(2) A PDF document named `<your_name>_task1.pdf` which is generated from your Jupiter Notebook source file, and presents clear and accurate explanation of your implementation and results. A poor presentation of results gains less marks in this task.

(Continue on next page)

Task 2

(3.5 marks)

Dataset: Webpage links (gr0.epa)
Source: http://www.cs.cornell.edu/courses/cs685/2002fa/data/gr0.epa

Dataset information
This dataset contains about 4800 webpages and their hyperlinks. The records with the "n" flag contain (unique) IDs and URLs of webpages. Those with the "e" flag represent hyperlinks between webpages. Essentially, this dataset models a "mini-WWW".

Objective
The objective of this task is to implement a PageRank algorithm (see the lecture slides of "**Graph Analytics**" in Week 11) to rank the webpages in the dataset.

Takes requirements
(i) Apply random teleports with probability 1.0 to dead ends in the web graph.
(ii) The transition matrix of the web graph is represented as a 4×4 block matrix and the (evolving) rank vector is represented as a 4×1 block matrix, where block matrices in this task refer to data structures of pyspark.mllib.linalg.distributed.BlockMatrix.
(iii) Use a teleport constant $\beta = 0.85$.
(iv) Set the initial rank vector to a uniform vector.
(v) Set the error parameter ε to 0.005.
(vi) Iterate the computation of the rank vector 20 times or until the rank vector converges (with the error parameter ε), whichever comes first.
(vii) Return the ranking scores of the first 20 webpages (i.e., webpages from ID 0 to ID 19).

Note: Spark's BlockMatrix supports matrix addition and multiplication (see its API doc). However, you may need to convert the rank vector (as a BlockMatrix) to other Spark data structure such as RDD in order to implement some other operations.

Deliverables
(1) A Jupiter Notebook source file named `<your_name>_task2.ipybn` which contains your implementation source code in Python.
(2) A PDF document named `<your_name>_task2.pdf` which is generated from your Jupiter Notebook source file, and presents clear and accurate explanation of your implementation and results. A poor presentation of results gains less marks in this task.

Task 3

(3.5 marks)

This task is related to the topic of "Stream Data Mining" in Week 12. The recording of Week 12 lab class in Moodle may help you to understand this task.

Note: Although it is recommended to directly use Spark MLlib's function to complete this task, you are allowed to use other libraries/modules/packages or program this streaming k-means clustering from the scratch by yourself to complete the task.

1. Generate three datasets in a two-dimensional space. Each dataset shall consist of at least three clusters, each of which has enough data. You can use "numpy.random.multivariate_normal" (see the hyperlink below) to create these clusters. Other functions can also to be used. Plot each dataset in one figure separately.

https://numpy.org/doc/stable/reference/random/generated/numpy.random.multivariate_normal.html

2. For each of the three datasets: turn it into a stream of data and apply Spark MLlib's Streaming k-means to obtain the cluster centres. Compare the obtained clusters with the true cluster centres you set by computing their Euclidean distances and visualising the obtained clusters in each iteration in each of the figure in the last step.

3. Concatenate the stream of the three databases in the order of "stream of dataset 1" followed by "stream of dataset 2" and then followed by "stream of dataset 3." Apply Spark MLlib's Streaming k-means to obtain the cluster centres. Visualise the obtained clusters in each iteration in a new figure to see if they can reflect the underlying change of the stream data from dataset 1, through dataset 2, to dataset 3.

Deliverables

(1) A Jupiter Notebook source file named `<your_name>_task3.ipybn` which contains your implementation source code in Python.
(2) A PDF document named `<your_name>_task3.pdf` which is generated from your Jupiter Notebook source file, and presents clear and accurate explanation of your implementation and results. A poor presentation of results gains less marks in this task.

--- END ---