BIG DATA PROCESSING

ASSIGNMENT 2: SPARK STREAMING,
SPARK STRUCTURED STREAMING,
ADDITIONAL SPARK LIBRARIES AND
ADDITIONAL TRANSPORTATION DATASETS.

<u>BACKGROUND.</u>

It's been already a few weeks since you started your short-term internship in the Big Data Engineering Department of the start-up OptimiseYourJourney, which will enter the market next year with a clear goal in mind: "*leverage Big Data technologies for improving the user experience in transportation*". Your contribution in Assignment 1 has proven the potential OptimiseYourJourney can obtain by using Spark Core and Spark SQL to analyse public transportation datasets as **Dublin Bus GPS sample data from Dublin City Council (Insight Project)**: https://data.gov.ie/dataset/dublin-bus-gps-sample-data-from-dublin-city-council-insight-project

OptimiseYourJourney

In the department meeting that has just finished your boss was particularly happy, again.

- First, she thinks this very same Dublin Bus dataset provides a great opportunity to explore the potential of Spark Streaming and Spark Structured Streaming in performing real-time data analysis. To do so, she asks you to adapt the exercises of the previous assignment for their application to these new technologies.
- Second, she is curious about the possibilities other Spark libraries, especifially devoted to Graph and Machine Learning algorithms, would offer when applied to this Dublin Bus Dataset. To do so, she asks you to write a short report about it.
- Third, she is curious about other publicly available transportation-related datasets (they can be focused on cars, buses, trains, taxis, bikes, scooters, etc). To do so, she asks you to write a short report about it.

DATASET.

Each real-time dataset you will be dealing with contains a number of files <'f1.csv', 'f2.csv', 'f3.csv', fn.csv'> and represents n batches (of 1 file each) arriving over time for their real-time data analysis. The datasets are provided to you in the folder my_dataset of **Canvas => 5_Assignments => A02.zip**.

As in Assignment 1, each row of an Assignment 2 dataset file contains the following fields:
Date , Bus_Line , Bus_Line_Pattern , Congestion , Longitude , Latitude , Delay , Vehicle , Closer_Stop , At_Stop

- **(00) Date**
  - A String representing the date of the measurement with format <%Y-%m-%d %H:%M:%S>
  - Example: "2013-01-01 13:00:02"
- **(01) Bus_Line**
  - An Integer representing the bus line.
  - Example: 120
- **(02) Bus_Line_Pattern**
  - A String identifier for the sequence of stops scheduled in the bus line (different buses of the same bus line can follow different sequence of stops in different iterations).
  - Example: "027B1001" (it can also be empty "").
- **(03) Congestion**
  - An Integer representing whether the bus is at a traffic jam (No => 0 / Yes => 1) .
  - Example: 0
- **(04) Longitude**
  - A Float representing the longitude position of the bus.
  - Example: -6.269634
- **(05) Latitude**
  - A Float representing the latitude position of the bus.
  - Example: 53.360504
- **(06) Delay**
  - An integer representing the delay of the bus with respect to its schedule (measured in seconds). It is a negative value if the bus is ahead of schedule.
  - Example: 90.
- **(07) Vehicle**
  - An integer identifier for the bus vehicle.
  - Example: 33304.
- **(08) Closer_Stop**
  - An integer identifier for the closest bus stop.
  - Example: 7486.
- **(09) At_Stop_Stop**
  - An integer representing whether the bus vehicle is at the bus stop right now (i.e., stopping at it for passengers to hop on / hop off). (No -> 0 and Yes -> 1)
  - Example: 0.

TASKS / EXERCISES.

The tasks / exercises to be completed as part of the assignment are described in the next pages of this PDF document.

- The following exercises are placed in the folder **my_code:**
    1. A02_ex1_spark_streaming.py
    2. A02_ex1_spark_structured_streaming.py
    3. A02_ex2_spark_streaming.py
    4. A02_ex2_spark_structured_streaming.py
    5. A02_ex3_spark_streaming.py
    6. A02_ex3_spark_structured_streaming.py
    7. A02_ex4_spark_streaming.py
    8. A02_ex4_spark_structured_streaming.py

    - **Each exercise is worth 7.5 marks.**

    Rules:

    - On each exercise, your task is to complete the function **my_model** of the Python program. This function is in charge of specifying the Spark Job performing the real-time data analysis.
    - When programming my_model, you can create and call as many auxiliary functions as you need.
    - Do not alter any of the other functions provided: get_source_dir_file_names, streaming_simulation, create_ssc and my_main.
    - Do not alter the parameters passed to the function my_model.
    - The entire work must be done "within Spark":
        - The function my_model must start with a creation operation textFileStream or readStream. These operations track the arrival of the batch files and load their content to Spark Streaming and Spark Structured Streaming, respectively.
        - The function my_model must finish with an action operation pprint or writeStream printing by the screen the result of the Spark Streaming / Spark Structured Streaming job.
        - The function my_model must not contain any other action operation other than the pprint or writeStream appearing at the very end of the function.

- The following exercises are placed in the folder **my_reports:**
    9. A02_report_additional_spark_libraries.odt
    10. A02_report_additional_transportation_dataset.odt

    - **Each report is worth 20 marks.**
    - Each report must contain a maximum of 1,000 words.

RUBRIC.

Exercises 1-8.

- 20% of the marks => Complete attempt of the exercise (even if it does not lead to the right solution or right format due to small differences).
- 40% of the marks => Right solution and format (following the aforementioned rules) for the "Small Dataset".
- 40% of the marks => Right solution and format (following the aforementioned rules) for any "Additional Dataset" test case we will generate. The marks will be allocated in a per test basis (i.e., if 4 extra test are tried, each of them will represent 10% of the marks).

Exercise 9.

- 25% of the marks => Originality.
- 25% of the marks => Relevance.
- 50% of the marks => Viability.

Exercise 10.

- 25% of the marks => Brief Description and Main Characteristics.
- 25% of the marks => Relevance.
- 50% of the marks => Viability.

TEST YOUR SOLUTIONS.

- The folder **my_results** contains the expected results for each exercise.
- Moreover, the subfolder **my_results/check_results** allows you to see if your code is producing the expected output or not.
  - The files **test_checker_spark_streaming.py** (resp. **test_checker_spark_structured_streaming.py**) need two files and compares whether their content is equal or not.
  - When you have completed one exercise (e.g., A02_ex1_spark_streaming.py), create a file with your result in the folder **my_results/check_results/Student_Solutions/**
  - Open the file **test_checker_spark_streaming.py** (resp. **test_checker_spark_structured_streaming.py** if the exercise you completed was A02_ex*_spark_structured_streaming.py) and edit the lines 141 and 142 (resp. 212 and 213) with the names of your file and the one you are comparing it against.
  - Run the program **test_checker_spark_streaming.py** (resp. **test_checker_spark_structured_streaming.py**).
    It will tell you whether your output is correct or not.

Main Message

Use the program **test_checker_spark_streaming.py** and **test_checker_spark_structured_streaming.py** to ensure that all your exercises produce the expected output (and in the right format!).

SUBMISSION DETAILS / SUBMISSION CODE OF CONDUCT.

Submit to Canvas by the 5th of December, 11:59pm.

- Submissions up to 1 week late will have 10 marks deducted.
- Submissions up to 2 weeks late will have 20 marks deducted.

On submitting the assignment you adhere to the following declaration of authorship. If you have any doubt regarding the plagiarism policy discussed at the beginning of the semester do not hesitate in contacting me.

Declaration of Authorship

I, ___YOUR NAME___, declare that the work presented in this assignment titled 'Assignment 2: Spark Streaming, Spark Structured Streaming, Additional Spark Libraries & Additional Transportation Datasets' is my own. I confirm that:

- This work was done wholly by me as part of my Msc. in Artificial Intelligence, my Msc. in Software Architecture and Design, my MSc. in Cloud Computing or my Msc. in Cyber Security at Munster Technological University.
- Where I have consulted the published work and source code of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this assignment source code and report is entirely my own work.

EXERCISE 1.

In Assignment 1, Exercise 1 was originally applied on Spark Core and Spark SQL. Now, in Assignment 2, we adapt the exercise to a real-time context for its application on Spark Streaming and Spark Structured Streaming. A full description of the original Exercise 1 can be found in the PDF description of Assignment 1 (pages 7-10). In the next sections:

- We provide the formal definition for the real-time version of the exercise.
  - We include any potential variation w.r.t. the original version of the problem due to limitations in the functionality of Spark Streaming and Spark Structured Streaming.
- We provide an example of a real-time dataset and the solution to be computed for it by Spark Streaming and Spark Structured Streaming.

EXERCISE: FORMAL DEFINITION

Given a program passing by parameters:

- The latitude upper bound for an area of interest, or "north" (e.g., 53.3702027)

- The longitude upper bound for an area of interest, or "east" (e.g., -6.2043634)

- The latitude lower bound for an area of interest, or "south" (e.g., 53.3343619)

- The longitude lower bound for an area of interest, or "west" (e.g., -6.2886331)

- A list of hours "hours_list" (e.g., ["07", "08", "09"])

and a dataset

- Containing several batches/files arriving over time (e.g., < f1.csv, f2.csv, f3.csv, f4.csv >)

your task is to:

- Treat each batch individually (independent from the other batches) and compute per individual batch:

- Spark Streaming: Compute the percentage of bus measurements withing the area of interest reporting congestion. Consider only the bus measurements of weekdays (you must discard any measurement taking place on a Saturday or Sunday). Compute the results aggregating them per each hour interval. Sort them by decreasing order in percentage of congestion measurements and, in case of tie, by increasing hour => Exactly the same as in Assignment 1.
- Spark Structured Streaming: Compute the percentage of bus measurements withing the area of interest reporting congestion. Consider only the bus measurements of weekdays (you must discard any measurement taking place on a Saturday or Sunday). Compute the results aggregating them per each hour interval. Do not sort the results of each batch => Different from what we did in Assignment 1.

The format of the solution computed after each new batch is processed must be:

- < (H1, P1, T1, C1), (H2, P2, T2, C2), ..., (Hn, Pn, Tn, Cn) >

where:

- Hi is a String representing the hour interval of the day (e.g, '09' - for [9am - 10am)).
- Pi is a Float representing the percentage of congestion measurements, with 2 decimal float point format (e.g., 3.54).
- Ti is an Integer representing the total number of measurements in this hour interval (e.g., 10000).
- Ci is an Integer representing the total number of measurements reporting congestion in this hour interval (e.g., 354).
- Spark Streaming:
    - < (H1, P1, T1, C1), (H2, P2, T2, C2), ..., (Hn, Pn, Tn, Cn) > are sorted by decreasing order of Pi. Any tie in percentage is broken by sorting the entries by increasing Hi order.
- Spark Structured Streaming:
    - < (H1, P1, T1, C1), (H2, P2, T2, C2), ..., (Hn, Pn, Tn, Cn) > are not sorted in any particular order.

EXAMPLE – SMALL DATASET

The folder “my_dataset/A02_ex1_micro_dataset_1” contains an example dataset containing 4 batches/ files

- < f1.csv, f2.csv, f3.csv, f4.csv >.

Given the aforementioned dataset and program parameters the files of the folder “my_result”:

- A02_ex1_spark_streaming.txt => Solution for Spark Streaming
- A02_ex1_spark_structured_streaming.txt => Solution for Spark Structured Streaming

Once again, please note each batch is treated individually (independent from the other batches):

- Results for Batch 1 contain the bus lines of < f1.csv >.
- Results for Batch 2 contain the bus lines of < f2.csv >.
- Results for Batch 3 contain the bus lines of < f3.csv >.
- Results for Batch 4 contain the bus lines of < f4.csv >.

EXERCISE 2.

In Assignment 1, Exercise 2 was originally applied on Spark Core and Spark SQL. Now, in Assignment 2, we adapt the exercise to a real-time context for its application on Spark Streaming and Spark Structured Streaming. A full description of the original Exercise 2 can be found in the PDF description of Assignment 1 (pages 11-15). In the next sections:

- We provide the formal definition for the real-time version of the exercise.
    - We include any potential variation w.r.t. the original version of the problem due to limitations in the functionality of Spark Streaming and Spark Structured Streaming.
- We provide an example of a real-time dataset and the solution to be computed for it by Spark Streaming and Spark Structured Streaming.

EXERCISE: FORMAL DEFINITION

Given a program passing by parameters:

- The bus vehicle "vehicle_id" (e.g., 33145)

- The concrete day being selected "day_picked" (e.g., "2013-01-02")

- The delay threshold or "delay_limit" (e.g., 60)

and a dataset

- Containing several batches/files arriving over time (e.g., < f1.csv, f2.csv, f3.csv, f4.csv >)

your task is to:

- Treat consecutively pairs of batches individually (i.e., < Batch 1 >, < Batch 1, Batch 2>, < Batch 2, Batch 3>, ..., < Batch n-1, Batch n> and compute for them:

- Spark Streaming: Compute the timetable of vehicle_id at day_picked, including the arrival time for all bus stations it stops at during the day, in chronological order. For each of them, include the station and line being served, and use delay_limit to determine whether the bus was on schedule when arriving at the station => Exactly the same as in Assignment 1.
- Spark Structured Streaming: Compute the timetable of vehicle_id at day_picked. Contiguous measurements cannot be identified nor discarded. Therefore, include all measurements where the bus is stopped at a station, in chronological order. For each of them, include the station and line being served => Different from what we did in Assignment 1.

The format of the solution computed must be:

- < (L1, S1, T1, O1), (L2, S2, T2, O2), ..., (Ln, Sn, Tn, On) >

where:

- Li is an Integer representing the ID of the line being served when arriving at the station (e.g., 50).
- Si is an Integer representing the ID of the station (e.g., 279).
- Ti is a String representing the arrival time at the station (e.g., "08:09:00").
- Oi in an Integer representing whether the bus was on schedule (1) or not (0) - (e.g., 0).
- The tuples (Li, Si, Ti, Oi) are sorted by increasing order of Ti.

<u>**EXAMPLE – SMALL DATASET**</u>

The folder “my_dataset/A02_ex2_micro_dataset_1” contains an example dataset containing 4 batches/ files

- < f1.csv, f2.csv, f3.csv, f4.csv >.

Given the aforementioned dataset and program parameters the files of the folder “my_result”:

- A02_ex2_spark_streaming.txt => Solution for Spark Streaming
- A02_ex2_spark_structured_streaming.txt => Solution for Spark Structured Streaming

Once again, please note <u>each consecutively pairs of batches are treated individually</u>:

- Results for Batch 1 contain the bus lines of < f1.csv >.
- Results for Batch 2 contain the bus lines of < f1.csv, f2.csv >.
- Results for Batch 3 contain the bus lines of < f2.csv, f3.csv >.
- Results for Batch 4 contain the bus lines of < f3.csv, f4.csv >.

EXERCISE 3.

In Assignment 1, Exercise 3 was originally applied on Spark Core and Spark SQL. Now, in Assignment 2, we adapt the exercise to a real-time context for its application on Spark Streaming and Spark Structured Streaming. A full description of the original Exercise 3 can be found in the PDF description of Assignment 1 (pages 16-19). In the next sections:

- We provide the formal definition for the real-time version of the exercise.
    - We include any potential variation w.r.t. the original version of the problem due to limitations in the functionality of Spark Streaming and Spark Structured Streaming.
- We provide an example of a real-time dataset and the solution to be computed for it by Spark Streaming and Spark Structured Streaming.

EXERCISE: FORMAL DEFINITION

Given a program passing by parameters:

- The start time for the time window "current_time" (e.g., "2013-01-07 06:30:00")

- The length of the time window "seconds_horizon" (e.g., 1800)

and a dataset

- Containing several batches/files arriving over time (e.g., < f1.csv, f2.csv, f3.csv, f4.csv >)

your task is to:

- Treat each batch individually (independent from the other batches) and compute per individual batch:

- Spark Streaming: Compute the bus station(s) with highest amount of bus vehicles stopping during the time window. For a given bus station, compute a list with the different bus vehicles stopping at it (if a bus stops twice or more at the station during the time window, you only count it once) => Exactly the same as in Assignment 1.
- Spark Structured Streaming: Compute the the amount of different bus vehicles stopping at each station (if a bus stops twice or more at the station during the time window, you only count it once) => Different from what we did in Assignment 1.

The format of the solution computed must be:

- < (S1, L1), (S2, L2), ..., (Sn, Ln) >

where:

- Si is an Integer representing the ID of the station (e.g., 279).
- Spark Streaming:
    - Li is an Integer list representing the bus vehicles IDs stopping at it during the time window, sorted in increasing order (e.g., [33145, 34000, 35000]).
    - In case multiple stations have the very same amount of max vehicles, sort them in increasing bus station ID order.
- Spark Structured Streaming:
    - Li is an Integer with the amount of bus vehicles stopping at this station.

EXAMPLE – SMALL DATASET

The folder "my_dataset/A02_ex3_micro_dataset_1" contains an example dataset containing 4 batches/ files

- < f1.csv, f2.csv, f3.csv, f4.csv >.

Given the aforementioned dataset and program parameters the files of the folder "my_result":

- A02_ex3_spark_streaming.txt => Solution for Spark Streaming
- A02_ex3_spark_structured_streaming.txt => Solution for Spark Structured Streaming

Once again, please note each batch is treated individually (independent from the other batches):

- Results for Batch 1 contain the bus lines of < f1.csv >.
- Results for Batch 2 contain the bus lines of < f2.csv >.
- Results for Batch 3 contain the bus lines of < f3.csv >.
- Results for Batch 4 contain the bus lines of < f4.csv >.

EXERCISE 4.

In Assignment 1, Exercise 4 was originally applied on Spark Core and Spark SQL. Now, in Assignment 2, we adapt the exercise to a real-time context for its application on Spark Streaming and Spark Structured Streaming. A full description of the original Exercise 4 can be found in the PDF description of Assignment 1 (pages 20-24). In the next sections:

- We provide the formal definition for the real-time version of the exercise.
    - We include any potential variation w.r.t. the original version of the problem due to limitations in the functionality of Spark Streaming and Spark Structured Streaming.
- We provide an example of a real-time dataset and the solution to be computed for it by Spark Streaming and Spark Structured Streaming.

EXERCISE: FORMAL DEFINITION

Given a program passing by parameters:

- The bucket or kilometre range interval size (measured in kilometres), or "bucket_size" (e.g., = 50)

- The maximum speed accepted (measured in meters per second), or "max_speed_accepted" (e.g., 28.0)

- The day selected, or "day_picked" (e.g., "2013-01-07")

and a dataset

- Containing several batches/files arriving over time (e.g., < f1.csv, f2.csv, f3.csv, f4.csv >)

your task is to:

- Aggregate the batches as they appear, computing each time a new batch appears:

- Spark Streaming: Compute the histogram for the bus vehicle fleet on a given day => Exactly the same as in Assignment 1.
- Spark Structured Streaming: Compute the aggregated amount of different bus stations each bus vehicle stops at (if a bus stops twice or more at the station during the aggregated time window, you only count it once) => Different from what we did in Assignment 1.

The format of the solution computed must be:

- Spark Streaming:
    - < (I1, R1, N1), (I2, R2, N2), ..., (In, Rn, Nn) >
    - where:
    - Ii is an Integer representing the index or bucket_ID (e.g., 0).
    - Ri is a String representing the kilometre range interval for the bucket (e.g., 0-50).
    - Ni is an integer representing the number of bus vehicles traversing a distance within the bucket (e.g., 5).

- Spark Structured Streaming:
    - < (V1, N1), (V2, N2), ..., (Vn, Nn) >
    - where:
    - Vi is an Integer representing the vehicle ID (e.g., 35000).
    - Ni is the aggregated amount of different bus stations each bus vehicle stops at.
    - < (V1, N1), (V2, N2), ..., (Vn, Nn) > are sorted by increasing order of Ni. Do not worry about breaking ties.

EXAMPLE – SMALL DATASET

The folder "my_dataset/A02_ex4_micro_dataset_1" contains an example dataset containing 4 batches/ files

- < f1.csv, f2.csv, f3.csv, f4.csv >.

Given the aforementioned dataset and program parameters the files of the folder "my_result":

- A02_ex4_spark_streaming.txt => Solution for Spark Streaming
- A02_ex4_spark_structured_streaming.txt => Solution for Spark Structured Streaming

Once again, please note the batch accumulated distance covered (Spark Streaming) or number of different bus stations stopped at (Spark Structured Streaming) by each vehicle:

- Results for Batch 1 contain the distance covered / different stations stopping at for all measurements of < f1.csv >.
- Results for Batch 2 contain the distance covered / different stations stopping at for all measurements of < f1.csv, f2.csv >.
- Results for Batch 3 contain the distance covered / different stations stopping at for all measurements of < f1.csv, f2.csv, f3.csv >.
- Results for Batch 4 contain the distance covered / different stations stopping at for all measurements of < f1.csv, f2.csv, f3.csv, f4.csv >.

EXERCISE 5.

Besides the Spark libraries covered in this semester < Spark Core, Spark SQL and Spark Real-Time Libs (Spark Streaming & Spark Structured Streaming) > Spark has also:

- A library especifially devoted to Graph algorithms
  - Spark GraphX (for working with RDDs)
  - Spark GraphFrames (for working with DataFrames)
- A library especifially devoted to Machine Learning algorithms
  - Spark MLlib (for working with RDDs)
  - Spark ML (for working with DataFrames)

Write a report of up to 1,000 words where you present and discuss:

- A novel exercise to be included in the data analysis of the Dublin Bus dataset involving the Spark Graph and/or Machine Learning libraries.

There is no need to implement the new exercise, you just need to discuss it in terms of:

- Its originality - It has to be different from the 4 exercises proposed in Assignments 1 and 2.
- Its relevance - Include a potential use-case derived from the exercise you are proposing.
- Its viability:
  - Do not implement the exercise, but briefly discuss in natural language (English and/or psudocode) the main steps that would be needed so as to implement it.
  - Include in the discussion whether, if you had to implement it, you would choose to implement it using the library version for working with RDDs or DataFrames. Justify your selection.
  - Position the new exercise in terms of difficulty with respect to the other four exercises proposed in this assignment.

EXERCISE 6.

Both Assignment 1 and Assignment 2 have had as reference the public transportation dataset **Dublin Bus GPS sample data from Dublin City Council (Insight Project)**: https://data.gov.ie/dataset/dublin-bus-gps-sample-data-from-dublin-city-council-insight-project

However, there are many other publicly available transportation-related datasets out there (they can be focused on cars, buses, trains, taxis, bikes, scooters, etc).

Write a report of up to 1,000 words where you present and discuss:

- A publicly available transportation-related dataset (different from the Dublin Bus dataset).

Compare and contrast the new dataset w.r.t. the Dublin Bus dataset, including:

- A brief description of the dataset.
- Its main characteristics: URL for accessing to it, size, format of the data, etc.
- Its relevance - Include a potential use-case derived from the dataset you are proposing that cannot be achieved with the Dublin Bus Dataset.
- Its viability:
    - ◦ Do not implement the use-case, but briefly discuss in natural language (English and/or psudocode) the main steps that would be needed so as to implement it.
    - ◦ Include in the discussion the Spark libraries you will use in case you have had to implement it. Justify your selection.
    - ◦ Position the new use-case in terms of difficulty with respect to the other exercises proposed in Assignment 1 and Assignment 2.