COMP60711 - DATA ENGINEERING

Coursework Description

Marking Scheme and Model Answers

Week 2

Coursework Description

1. Data Description

You have been given access to two road-traffic data files in CSV format, **'rawpvr_2018-02-01_28d_1083 TueFri.csv'** and '**rawpvr_2018-02-01_28d_1415 TueFri.csv**'. Each file contains observations collected via Inductive Loops sensors planted on a particular site of Chester Road in the city of Manchester. For example, file **'rawpvr_2018-02-01_28d_1083 TueFri.csv'** contains observations collected from site 1083, while file '**rawpvr_2018-02-01_28d_1415 TueFri.csv'** contains observations collected from site 1415. The observations were collected during the month of February 2018, one of the busiest winter months of the year. Note that each row of the file contains one observation, i.e., the properties associated with one detected vehicle. As a consequence, if you count the total number of records with the following timestamp (06/02/2018), you are able to estimate the total volume of traffic (i.e., the total vehicle count) on the 6th of February 2018.

Both data files present the same structure, composed of the following attributes or properties:

- ***Date*** is a timestamp containing day of the month and year and time of the day when a vehicle was detected, with the following format: dd/mm/yyyy HH:MM:SS.
- ***Lane*** is an identifier of a given lane of the road (a road may have multiple lanes and each lane has a unique identifier).
- ***Lane Name*** is the name given to a particular lane of the road. Each lane has a unique name.
- ***Direction*** identifies the direction followed by a road lane (e.g., North, South, etc.). Different lanes may follow the same direction.
- ***Direction Name*** is the name of the direction followed by a lane of the road.
- ***Speed (mph)*** is the speed with which the detected vehicle was moving at the time it was detected.
- ***Headway (s)*** is the time distance between two consecutive vehicles following the same route. More precisely, it is the time distance between the front bumper of one vehicle and the front bumper of the vehicle behind it.
- ***Gap (s)*** is also a time distance between two consecutive vehicles following the same route, but it indicates the time distance between the rear bumper of one vehicle and the front bumper of the vehicle behind it.
- ***Flags*** is a number that identifies the day of the week when a vehicle was detected.
- ***Flag Text*** is the text description of the day of the week when a vehicle was detected.

2. Developing the Coursework

The coursework is composed of a list of tasks, divided into three sub-lists. This is the second one, described as follows:

Sub-list 2: This is the second sub-list of tasks and contains three tasks, described in Section 3.

When developing the coursework, you should do the following:

1. Use file '**rawpvr_2018-02-01_28d_1083 TueFri.csv**' for all tasks, unless you are explicitly told in the task to use both files, **'rawpvr_2018-02-01_28d_1083 TueFri.csv'** and '**rawpvr_2018-02-01_28d_1415 TueFri.csv**'.

2. When asked to use a technology of your choice to develop your work, use the one you are most familiar with from the suggested ones in Section 4, to avoid steep learning curves.

3. You should upload all your programming solutions into your **Gitlab COMP60711_Part_1 Project**, and answer the essay questions (which require text answers from you) in the **Blackboard Test** associated with this piece of coursework, titled "*60711-Lab1-S-CW1*".

4. To be able to access your Gitlab COMP60711_Part_1 Project, do the following so that an account is automatically created for you:
    a. **Log in to Gitlab using your University username and password.**

5. In more detail, make sure you upload to your Gitlab COMP60711_Part_1 project ALL the Python code you developed for the tasks that require programming, described in Section 3, naming your file as follows: '*CW1.py'.* On Gitlab, you will find the file template with a code skeleton into which you should insert your code, as described in the provided instructions.

    Also, provide explanations of your Python code in the form of comments that you place within the code itself. You can add a paragraph of comment on top of one line or a group of lines of code that you deem to be associated (by functionality).

    Identify and justify any data preparation steps.

    For each comment, use a maximum of *50 words* of text. If the code plots a graph, then make sure that the plotted graph is added to your '*Figures_SolutionCourseworkTasks2to4'* file (see item 6 for details about this file), and that the graph/plot is referred to in the code comments, using the its distinct number (see instructions in item 6).

6. You should export and paste into a single PDF document any relevant graphs/plots you generate for the tasks described in Section 3 of this document, giving to each graph/plot a distinct number, so that you are able to refer to an individual plot/graph from any code comment of any of the text answers you will be submitting via the Blackboard Test titled "*60711-Lab1-S-CW1*". Finally, upload this PDF document (with the graphs/plots) into your Gitlab repository, naming it '*Figures_SolutionCourseworkTasks2to4'.*

7. Note that all your essay-like, text answers (for the tasks described in Section 3 of this document) must be written in and submitted via the Blackboard Test titled "*60711-Lab1-S-CW1*", accessible through the course unit's Blackboard space, in folder 'Coursework Material for PART 1 of the Course (Weeks 1, 2 and 3)'. Despite the answers being written on Blackboard, you can refer to any plots/graphs you generated for the tasks in your answers, using the plot/graph number you have specified in file '*Figures_SolutionCourseworkTasks2to4'.*

8. To access the text field areas where your answers should be written, start Test "*60711-Lab1-S-CW1*". Once you have started the Test, you will see all questions relevant to this coursework.

   You must **manually save all your answers** all the time and especially before leaving the test and closing the relevant window browser, in order to be able to resume your coursework at a later time, until the deadline. Use a maximum of 800 words for each task.

   Also, to provide an answer to each of the questions in these tasks, please, START the test. You can interrupt your work in this test at any time to continue at a later time, provided that you MANUALLY SAVE the current draft of your answers without submitting it. If you submit it and begin the test again, you will not see again your previously attempted answers and will need to start from scratch!
   Therefore, you must submit your test only once.
   Once you have finished the test and checked your answers, then you can manually SUBMIT your answers.
   Make sure you SUBMIT your answers INSIDE the coursework DEADLINE.

9. **VERY IMPORTANT:** Note that the outcome of Tasks 2.I, 2.II, 2.III, and 3.I is a set of numeric results and, so, in the Python code you are going to develop for each of these tasks, you MUST use a Python ***dictionary*** to hold the numeric results. This is necessary to allow the marking of your code to be performed with success.

   Recall that, in Python, a dictionary is a built-in data structure for storing groups of objects. It consists of a mapping of key-value pairs, where each key is associated with a value.

10. After uploading your *'CW1.py'* file to Gitlab, as well as your '*Figures_SolutionCourseworkTasks2to4'* and '*2ndSolutionForTask3_I'* files, go to our course unit's site on Blackboard and click on the link titled "**COMP60711 Data Engineering Coursework Submission**" to submit your Gitlab project to Blackboard. Upon clicking on the link, through the provided interface, you should press the GitLab button on the Gradescope submission page (an application that the University uses to assess Gitlab projects) and select the repo to submit the work.

Do not ***arbitrarily*** discard/delete rows in the data files provided to you, unless explicitly required in the task question.

3. Tasks Description

Week_2	
Task_2	***For all tasks that require programming, make the task's solution available to us through Gitlab, as explained in Section 2. Also, follow instruction number 9 in Section 2, regarding the use of a Python dictionary to hold multiple results.*** In traffic analysis, it's standard procedure to develop profiles for various city roads before making critical decisions. These decisions could include determining the specific locations on the road to install traffic lights and optimising the timing settings for these lights. A basic profile of a road segment can be created by deriving descriptive data summarisation measures. To gain insight into the typical vehicle speed patterns observed at site 1083, provide a basic profile of vehicle speeds on the North lanes. This profile should be based on the following descriptive data summarisation measures: Range (R), 1st Quartile (Q1), 2nd Quartile (Q2), 3rd Quartile (Q3), and Interquartile Range (IQR), as follows: I. First, calculate each of these measures for each individual North lane (separately), focussing only on Tuesdays, as weekday, and time of the day between 09:00 am and 09:59:59 am. **[3 marks]** If you are curious about why we have selected Tuesday and the time period between 09:00 and 09:59:59 am, the explanation is straightforward. This selection is based on traffic patterns in most UK cities, where Tuesday is typically the busiest day of the week. Furthermore, the chosen time slot corresponds to one of the peak traffic periods - the morning rush hour. This is when parents have dropped off their children at school and are commuting to work. II. Second, do the same for each individual South lane, focussing on the same weekday and time of day. **[3 marks]** III. To enrich your profile of the road traffic around site 1083, calculate the same measures considering the traffic volume for each individual North lane, and for each individual South lane, considering the same day of the week and time of the day. **[6 marks]** IV. Before uploading your Python code to Gitlab, make sure you provide brief explanations/descriptions of the code in the form of comments that you place within the code itself. While doing this, identify in the code any data preparation steps you implemented, justifying why these steps are associated with the activity of data preparation and why these steps were necessary for accomplishing Tasks 2.I, 2.II and 2.III. Please, proceed to the Blackboard Test titled "*60711-Lab1-S-CW1*". Follow the instructions provided in the test to answer this task. Completing this will enable you to receive a mark for this task. **[5 marks]**

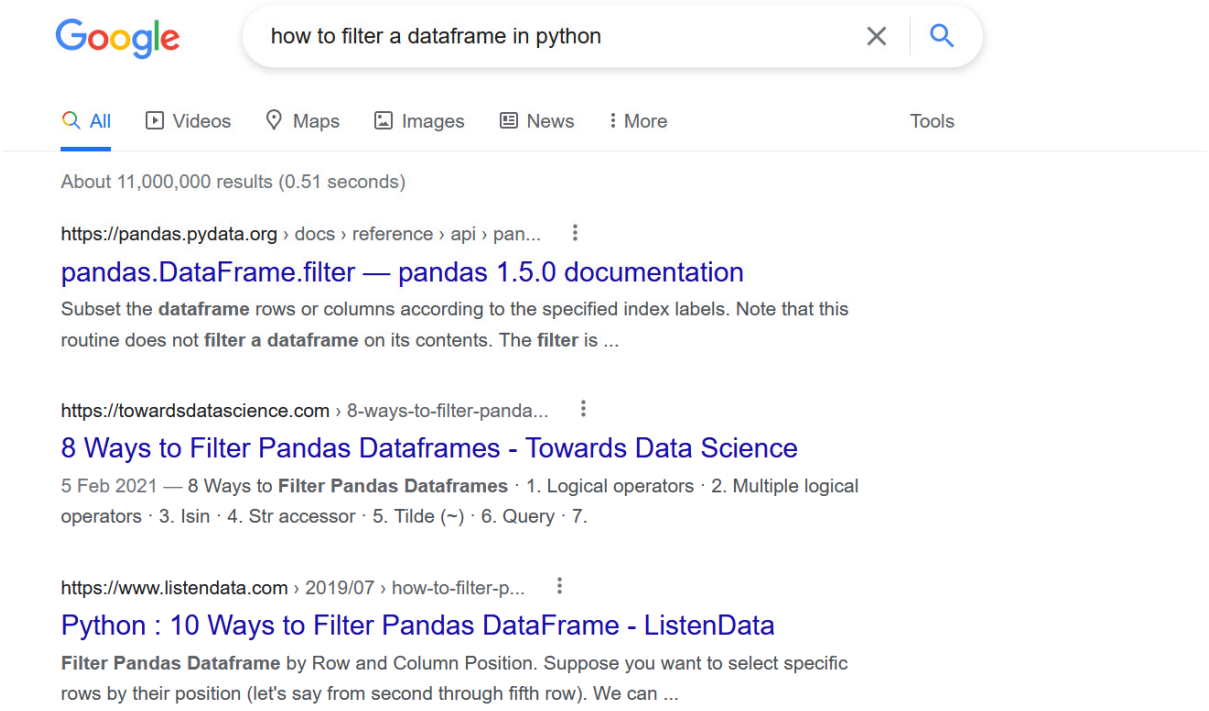
	After completing Tasks 2.I, 2.II, 2.III and 2.IV, you should be able to make observations, derive insights and draw conclusions from the profile you obtained. For that, provide answers to the following questions via the Blackboard Test specified in the instruction of Section 2, "*60711-Lab1-S-CW1*". V. Describe the insights that the five DDS measures can provide to a traffic data analyst in the context of this use case, without including the general definition of each measure. Instead, focus on what these measures reveal to a traffic analyst. **[6 marks]** VI. Describe the relevant information or insights for this use case that the five DDS measures fail to provide to a traffic data analyst. Discuss the impact that the absence of this information has on the data analysis, specifically in the context of this use case, which involves analysing traffic volume and speed. **[6 marks]** VII. Describe the conclusions/insights that can be drawn from the profile of the target road fragment. For this, consider commenting, for example, about speed and traffic volume for the North and South lanes. **[8 marks]**
Task_3	***For all tasks that require programming, make the task's solution available to us through Gitlab, as explained in Section 2. Also, follow instruction number 9 in Section 2, regarding the use of a Python dictionary to hold multiple results.*** By completing Task 2, you have built a minimal profile of the road fragment identified as site 1083. It is deemed to be minimal since it does not contain any information about road traffic at other times of the day or days of the week. Nonetheless, this information is useful and could be used, for example, to predict journey times for any car trip that includes site 1083 in its itinerary. I. To extend the profile of site 1083, calculate the average traffic volume per hour of the day for Tuesday and, separately, the same for Friday. For simplification purposes, consider only the times of the day between 07:00:00 to 23:59:59, ignoring vehicles detected between 00:00:00 and 06:59:59. Also, do not make distinctions between North and South lanes, considering all together in your calculation. **[7 marks]** II. Before uploading your Task 3.I Python code to Gitlab, make sure you provide brief explanations/descriptions of the code in the form of comments that you place within the code itself. For this task, there is NO need to justify data preparation steps. Then, please proceed to the Blackboard Test titled "*60711-Lab1-S-CW1*". Follow the instructions provided in the test to answer this question. Completing this will enable you to receive a mark for this task. **[5 marks]** III. From the results you obtained in Task 3.I, what were you able to observe (feel free to consider any aspects associated with the obtained profile/results, profiling techniques, etc.)? To provide an answer to this

	question, please, proceed to the Blackboard Test "*60711-Lab1-S-CW1*". Follow the instructions provided in the test to answer this task. Completing this will enable you to receive a mark for this task. **[8 marks]**
Task_4	***For all tasks that require programming, make the task's solution available to us through Gitlab, as explained in Section 2.*** You have been asked to develop the previous tasks using the Python programming language, which is one of the most popular languages for tabular data processing, adopted by programmers worldwide and in companies such as Netflix, Google and Amazon. However, not all data analysts possess programming skills. For those, alternative means are at disposal, including technology that have existed for decades, such as Database Management Systems and query languages, such as MySQL and SQL, as well as others that have more recently been developed, including OpenRefine and Knime, which offer Graphical User Interfaces to facilitate data profiling and preparation. Choosing a technology other than Python (from the suggested in Section 4), repeat *Task_3.I*, calculating the average traffic volume per hour of the day for Tuesday and, separately, the same for Friday. Again, for simplification purposes, consider only the times of the day between 07:00:00 to 23:59:59, ignoring vehicles detected between 00:00:00 and 06:59:59, and make no distinctions between North and South lanes. I. Make sure you upload into your Gitlab COMP60711_Part_1 project the step-by-step code/recipe you developed using the technology of your choice. Please, name it '*2ndSolutionForTask3_I*'. Also, describe in the "*60711-Lab1-S-CW1*" Blackboard Test text field the features of the technology you used, emphasising the ones that have made your work easier and the ones that made your work more difficult, in contrast to Python. In your description, you can make comments about functionality that is not offered within the technology, rendering you unable to satisfactorily complete *Task 3.I*. **[12 marks]**

4. Further Advice

The following list contains the PL and Data Manipulation/Preparation tools we suggest you use in the development of your coursework: *Python, Knime, OpenRefine and MySQL*. These are of easy access and free installation. You can access them from the machines in lab, or you can just download and install them in your machine.

If you are not familiar with the suggested programming language and/or tools, then you can search for commands in the Web, as shown below:



If using Python, for example, you will be interested in using commands from packages such as *pandas*, *numpy*, *datetime*, *os* and *calendar*, to handle *Date* related data types.

General tutorials for each of these can be found from the following links:

- **Knime** (https://www.knime.com/downloads/download-knime)
  Documentation: (https://docs.knime.com/)
  Tutorials: (https://www.youtube.com/watch?v=HEp9Cbql2hs,
  https://www.youtube.com/watch?v=5WAyOiIfHPg)

- **OpenRefine** (https://openrefine.org/download.html)
  (https://www.youtube.com/watch?v=WCRexQXYFrI),
  (https://www.youtube.com/watch?v=wGVtycv3SS0),
  (https://www.youtube.com/watch?v=wfS1qTKFQoI)

- **Python** (https://www.tutorialspoint.com/python/index.htm)

- **MySQL** (https://www.mysql.com/downloads/)
  (https://www.tutorialspoint.com/mysql/index.htm)