COMP20008 Project 1

V1.0: August 14, 2019

DUE DATE

The assignment is worth 20 marks, worth (20% of subject grade) **and is due Friday 6 September 2019**. Submission is via the LMS. Please ensure you get a submission receipt via email. If you don't receive a receipt via email, this means your submission has not been received and hence cannot be marked. Late penalty structure is described at the end of this document.

Introduction

Road fatalities in Victoria in the first half of 2019 increased by 50% compared with the same period in 2018. Across Australia, speeding was the single largest factor contributing to road fatalities. Over one third of these road fatalities occurred in capital cities.

As a data scientist, you would like to understand the patterns surrounding motorists speeding in Melbourne so that you can suggest targeted interventions. To that end, you are pointed to the *Traffic Count Vehicle Classification* dataset from the *City of Melbourne*. The dataset contains survey results showing the number of vehicles of different types that have traversed particular road segments at a particular time. It also includes the speed limit of the roads and data related to the speeds of vehicles travelling on the roads. You can see a full explanation of the dataset at the City of Melbourne website[1]

For this project, you will apply various wrangling skills including processing and visualisation techniques to make sense of the data. As a data scientist, you are also expected to be able to use library functions which are unfamiliar and which require you to consult additional documentation from resources on the Web. You are also expected to ensure that all of the visualisations you produce are effective communication tools. In particular, all axes should be labelled, legends included where appropriate, and all visualisations should include an informative title.

The project is to be coded in Python 3. Three relevant datasets can be downloaded from LMS:

- `traffic.csv`

[1] https://data.melbourne.vic.gov.au/Transport-Movement/Traffic-Count-Vehicle-Classification-2014-2017/qksr-hqee.

- roads.json
- special_traffic.csv (for visualisation stage only)

Stage 1: Understand the dataset (3/20 marks)

To begin, we would like to understand the characteristics of the dataset we are working with. We would also like to clean the dataset by removing malformed or missing entries that will make analysis more difficult.

Q 1.1 (1 mark)

Print the number of traffic survey entries, number of attributes, attribute names and their data types from `traffic.csv` dataset. The output of this step should look like

```
***
Q1.1
Number of traffic survey entries: #
Number of attributes: #

@ $
@ $
...
@ $
***
```

where # is the value you find, @ is an attribute name and $ is its datatype.

Q 1.2 (1 mark)

A number of survey entries detected no vehicles of any type. For some of these entries, a maximum_speed of '-' has been included. For others, the maximum speed is blank. These entries can cause problems when analysing data. Create a DataFrame `traffic` from `traffic.csv` dataset with all such entries removed and print the number of remaining traffic survey entries. Your output should look like this:

```
***
Q1.2
Number of remaining traffic survey entries: #
***
```

Q 1.3 (1 mark)

The *vehicle_class_1* attribute represents the number of short vechicles, such as cars, detected in a survey hour. Using `traffic` DataFrame (from Q1.2), what is the median value of *vehicle_class_1*? What is the highest *maximum_speed* detected across all survey entries? Your code should print out the results with the following format:

```
***
Q1.3
Median value of vehicle_class_1: #
Highest value of maximum_speed: #
***
```

where # is the value you find, **rounded to 1 decimal place**.

Stage 2: Data selection & manipulation (5/20 marks)

We are particularly interested in understanding traffic patterns across different types of roads. While the `traffic.csv` file does not contain any information about the type of road a survey entry is related to, this information is contained in the dataset `roads.json`. We can map the *road_segment* attribute from the `traffic` DataFrame with the *SegID* attribute in the `roads` JSON dataset to discover which type of road a particular survey entry is related to.

Q 2.1 (2 marks)

Using the `traffic` DataFrame as well as `roads.json`, add the attribute *StrType* from the JSON dataset to the `traffic` DataFrame. Print the first 3 rows.
The output of this step should look like

```
***
Q2.1
The first three rows of traffic DataFrame with the attribute StrType are:
@
@
@
***
```

where @ is the entire row in the DataFrame.

Q 2.2 (1 mark)

We would now like to understand which roads have the most serious incidents of speeding. Add a new attribute *max_speed_over_limit* to `traffic` DataFrame. This column should represent the difference between the *maximum_speed* and the *speed_limit* attributes; that is

max_speed_over_limit = *maximum_speed*- *speed_limit*

Print the first 3 rows. The output of this step should look like

```
***
Q2.2
The first three rows of traffic DataFrame with the new max_speed_over_limit attribute are:
@
@
```

```
@
***
```

where @ is the entire row in traffic DataFrame.

Q 2.3 (2 marks)

We are particularly concerned with instances of speeding on arterial roads, as arterial roads are the responsibility of VicRoads rather than local councils.

Create a new DataFrame, `arterials`, which contains only the survey entries from traffic that relate to Arterial roads. Group the survey results in your `arterials` DataFrame by *road_name*. Print the names of the three roads with the highest maximum *max_speed_over_limit*. (1 mark)

Comment on how you think this information could be useful for VicRoads. (1 mark)

The output of this step should look like

```
***
Q2.3
Three Arterial roads with the highest maximum max_speed_over_limit:
@: #
@: #
@: #
***
```

where @ is the name of the road and # is the maximum *max_speed_over_limit* for that road.

Stage 3: Visualisation and Clustering (11/20 marks)

We now start to get a sense of the data characteristics through various types of visualisation.

Q 3.1 Plotting groups (3 marks)

Using the `traffic` DataFrame, draw a the following two plots: (2 marks)

a A bar plot showing *suburb* (x-axis) versus mean *average_speed* (y-axis).

b A Tukey boxplot showing the distribution of *vehicle_class_1* (the number of short vehicles) within the traffic survey results.

Comment on what you can observe from this output. (1 mark)

Note: You may need to do additional data cleaning to handle other invalid survey entries before plotting the data

Q 3.2 Dimension reduction and visualisation (4 marks)

The special_traffic.csv dataset contains entries from traffic.csv with some added information. Although there are fewer entries than in the original traffic.csv dataset, there are still too many to visualise efficiently. For this set of visual analysis, we wish to create a DataFrame special_traffic by randomly sampling 1000 entries from the dataset special_traffic.csv. The attribute *idx* is the identifier of a traffic survey entry and *StrType* is the road type of the survey entry.

Using this special_traffic DataFrame and all attributes except for *StrType* and *idx* as features:

a Perform Principal Component Analysis to determine the first and second principal components of this dataset. Produce a scatter plot of the first 2 principal components, colouring the points by their *StrType* value. (1 mark)

b Interpret the scatter plot. (1 mark)

c Perform two VAT visual analyses: one on all features and the other on the two components from task Q 3.2.a (1 mark)

d Interpret the two VAT plots and explain the differences. (1 mark)

Q 3.3 Clustering and visualisation(4 marks)

Perform K-means clustering on all data in the special_traffic DataFrame using all attributes except for *StrType* and *idx* as features.

a Recall that the SSE (sum of squared errors) can be used to measure the quality of clustering. The SSE is the sum of distances of objects from their cluster centroids:

$$SSE = \sum_{i=1}^{k} \sum_{x \in c_i} distance(x, \overline{c_i})^2$$

Produce a plot of SSE vs the number of clusters k as you vary the number of clusters. Use the 'elbow method' to identify the optimal value of k from your plot. Is this expected given your previous results in Q3.2? Why or why not? (1.5 marks)

b Perform K-means clustering on the data for $k = 3$. Show the size of each cluster with a bar plot. (0.5 marks)

We are interested in knowing if the model groups data well based on the attribute *StrType.*

c Suggest a plot to help one visually evaluate the K-means model against the measurement statement. Justify your suggestion. (1 mark)

d Draw the suggested plot and explain your finding. (1 mark)

Marking scheme

Correctness (19 marks): For each of the questions, a mark will be allocated for level of correctness (does it provide the right answer, is the logic right), according to the number in parentheses next to each question. Note that your code should work for any data input formatted in the same way as `traffic.csv`, `roads.json`, and `special_traffic.csv`. E.g. If a random sample of 1000 records was taken from `traffic.csv:`, your code should provide a correct answer if this was instead used as the input.

Correctness will also take into account the readability and labelling provided for any plots and figures (plots should include title of the plot, labels/scale on axes, names of axes, and legends for colours symbols where appropriate).

Coding style (1 mark): A Mark will be allocated for coding style. In particular the following aspects will be considered:

- Formatting of code (e.g. use of indentation and overall readability for a human)
- Code modularity and flexibility. Use of functions or loops where appropriate, to avoid highly redundant or excessively verbose definitions of code.
- Use of Python library functions (you should avoid reinventing logic if a library function can be used instead)
- Code commenting and clarity of logic. You should provide comments about the logic of your code for each question, so that it can be easily understood by the marker.

Resources

The following are some useful resources, for refreshing your knowledge of Python, and for learning about functionality of pandas.

- Python tutorial
- Python beginner reference
- pandas 10min tutorial
- Official pandas documentation
- Official mathplotlib tutorials
- Python pandas Tutorial by Tutorialspoint
- pandas: A Complete Introduction by Learn Data Sci
- pandas quick reference sheet
- sklearn library reference
- NumPy library reference
- Python Data Analytics by Fabio Nelli (available via University of Melbourne sign on)

Submission Instructions

Via the LMS, submit a jupyter notebook (A template notebook ”notebook-for-answers.ipynb” is provided in the folder with the datasets) containing your Python 3 code.

Other

Extensions and Late Submission Penalties: If requesting an extension due to illness, please submit a medical certificate to the lecturer. If there are any other exceptional circumstances, please contact the lecturer with plenty of notice. Late submissions without an approved extension will attract the following penalties

- $0 < hourslate <= 24$ (2 marks deduction)
- $24 < hourslate <= 48$ (4 marks deduction)
- $48 < hourslate <= 72$: (6 marks deduction)
- $72 < hourslate <= 96$: (8 marks deduction)
- $96 < hourslate <= 120$: (10 marks deduction)
- $120 < hourslate <= 144$: (12 marks deduction)
- $144 < hourslate$: (20 marks deduction)

where *hourslate* is the elapsed time in hours (or fractions of hours).

This project is expected to require 20-25 hours work.

Academic Honesty

You are expected to follow the academic honesty guidelines on the University website
`https://academichonesty.unimelb.edu.au`

Further Information

A project discussion forum has also been created on the subject LMS. Please use this in the first instance if you have questions, since it will allow discussion and responses to be seen by everyone. There will also be a list of frequently asked questions on the project page.