Deakin University
School of IT
Unit Team: SIT742

Trimester 2, 2024
Assignment 2

SIT742: Modern Data Science

Extension Request Students with difficulty in meeting the deadline because of various reasons, must apply for an assignment extension no later than 5:30pm on 20/09/2024 (Friday). Apply via '`CloudDeakin`', the menu item '`Extension Request`' under the '`Assessment`' drop-down menu.

Academic Integrity All assignment will be checked for plagiarism, and any academic misconduct will be reported to unit chair and university.

Generative AI Deakin's Policy and advices on responsible usage of Generative AI in your studies: `https://www.deakin.edu.au/students/study-support/study-resources/artificial-intelligence`

Instructions

Assignment Questions

There are total **2** parts in the assessment task 2:

Part 1 The first part will focus on the data manipulation and pyspark skills which includes the `Data Acquisition`, the `Data Wrangling`, the `EDA` and `Spark`, the `modules and library` from **M03, M04**.

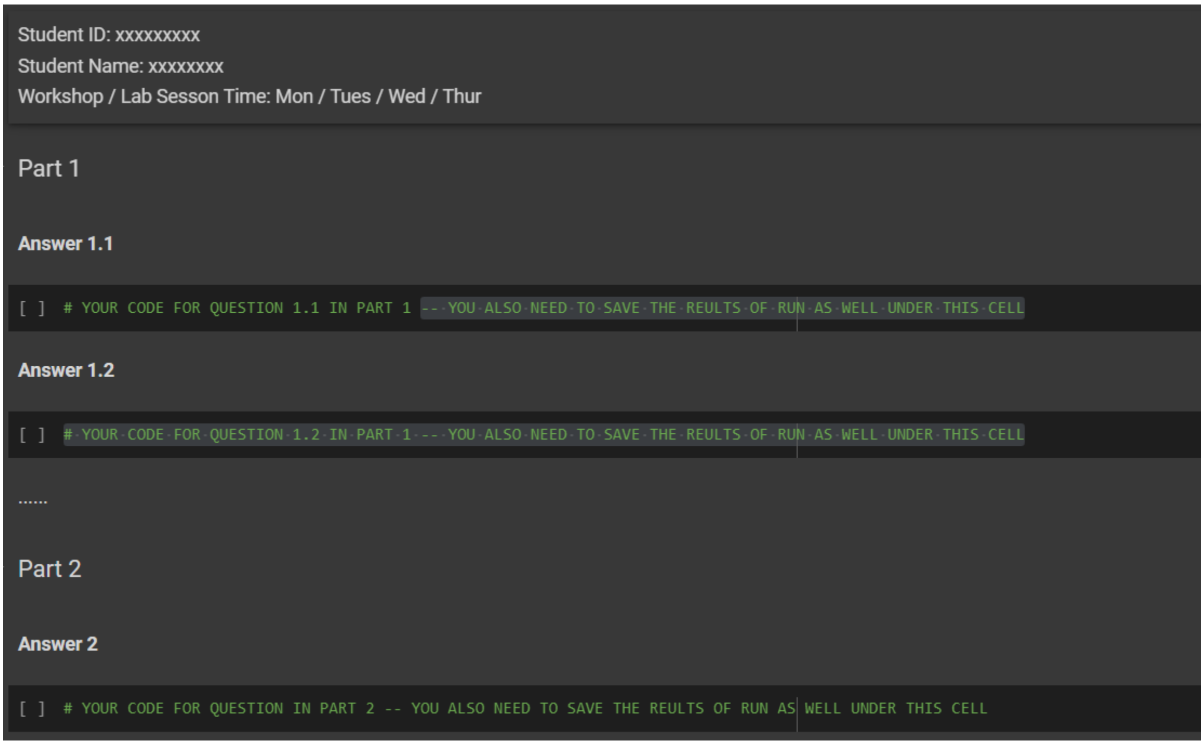
Part 2 The second part focus on more advanced data science skills with particular scenario. This part will require the knowledge covered in **M05**.

What to Submit?

There is no optional part for assignment 2. You (your group) are required to submit the following completed files to the corresponding *Assignment* (Dropbox) in *CloudDeakin*:

`SIT742Task2.ipynb` The completed notebook with `all the run-able code` for all requirements (part 1 and part2).

In general, you (your group) need to complete, **save** the results of running, download/export the notebook as a local file, and submit your **notebook** from Python platform such as `Google Colab`. You need to clearly list the answer for each question, and the expected format from your notebook will be like in Figure 1 (**One notebook** for each group).



Student ID: xxxxxxxxx
Student Name: xxxxxxxx
Workshop / Lab Sesson Time: Mon / Tues / Wed / Thur

Part 1

Answer 1.1

```
[ ]  # YOUR CODE FOR QUESTION 1.1 IN PART 1 -- YOU ALSO NEED TO SAVE THE REULTS OF RUN AS WELL UNDER THIS CELL
```

Answer 1.2

```
[ ]  # YOUR CODE FOR QUESTION 1.2 IN PART 1 -- YOU ALSO NEED TO SAVE THE REULTS OF RUN AS WELL UNDER THIS CELL
```

......

Part 2

Answer 2

```
[ ]  # YOUR CODE FOR QUESTION IN PART 2 -- YOU ALSO NEED TO SAVE THE REULTS OF RUN AS WELL UNDER THIS CELL
```

Figure 1: Notebook Format

`SIT742Task2report.pdf` You (group) are also required to write a report with your answer (code) and running results from `SIT742Task2.ipynb` for all the questions (`Part 1` and `Part 2`). You could make screenshot on your answer (code) and running results from `SIT742Task2.ipynb` and paste into the report. Please try to include the code comments, and results including plot images as well in the report, and make sure the code format such as Indentation keeps same as the ipynb notebook.

In this report (**one for each group**), you will also need to provide a clear explanation on your logic for solving each question (you could write explanation below your solution and results in the report). In the explanation, you will need to cover below parts: 1). why you decide to choose your solution; 2). are there any other solutions that could solve the question; 3). whether your solution is the optimal or not? why? The length of the explanation part for each question is limited below 100 words.

In the end of your report, you (group) also need to discuss below three points:

- How you and your team member collaborate on this assignment?
- What you have learned with your team member from the second assignment.
- What is the contribution of each the team member for finishing the second assignment.

`SIT742Task2video.avi` A video demonstration between 10 and 15 minutes, and the file format can be any common video format, such as '`MKV`', '`WMV`', '`MOV`' etc.

For your group, one important submission is a **short video** in which each of *You* (group members) orally present the solutions that you provide in the notebook and illustrate the running of code with the used logic. In the video, your group need to work together to discuss below three points:

- Which question(s) you have worked on and how did you collaborate with other team members.
- What is the logic behind the your solution on the question(s)? and is there any alternative optimized ways to resolve the question?
- What is your understanding of `Code collaboration`? How do you collaborate with your group in coding? What are the common tools/platform to support the `Code collaboration`?

Part I

Data Acquisition and Manipulation

There are **10** questions in this part, totalling **60** marks. Each of question is worth **5** marks. Additionally, the quality of your explanation in both the report and video will collectively be worth **10** marks.

You are recommended to use `Google Colab` to finish all the coding in the *code block cell*, and provide sufficient coding comments, and also save the result of running as well.

The (`transactionrecord.zip`) data used for this part could be found in **here**. You will need to use `spark` to read the unzipped (csv) data for starting. You could find the code on reading csv data with Spark from M04G.

Question 1.1

Using PySpark to do some of the data wrangling process, so that:

1.1.1 For the `'NA'` in `CustomerNo` columns, change it to `'-1'`.

1.1.2 Process the text in `productName` column, only alphabet characters left, and save the processed result to a new column `productName_process` and show the first 5 rows.

Question 1.2

Find out the revenue on each transaction date. In order to achieve the above, some wrangling work is required to be done:

1.2.1 Using `pyspark` to calculate the `revenue` (`price` * `Quantity`) and save as float format in pyspark dataframe to show the top 5 rows.

1.2.2 Transform the pyspark dataframe to `pandas` dataframe (named as `df`) and create the column `transaction_date` with date format according to Date. Print your `df` pandas dataframe with top 5 rows after creating the column `transaction_date`.

1.2.3 Plot the sum of `revenue` on `transaction_date` in a line plot and find out any immediate pattern / insight?

Question 1.3

Let's continue to analyse on the `transaction_date` vs `revenue`.

1.3.1 Determine which `workday` (day of the week), generates the most sales (plotting the results in a line chart with workday on averaged revenues).

1.3.2 Identify the name of product (column `productName_process`) that contributes the highest revenue on '`that workday`' (you need to find out from 1.3.1) and the name of product (column `productName_process`) that has the highest sales volume (sum of the `Quantity`), no need to remove negative quantity transactions.) on '`that workday`' (you need to find out from 1.3.1).

1.3.3 Please provide two plots showing the top 5 products that contribute the highest revenues in general and top 5 products that have the highest sales volumes in general.

Question 1.4

Which country generates the highest revenue? Additionally, identify the month in that country that has the highest revenue.

Question 1.5

Let’s do some analysis on the `CustomerNo` and their transactions. Determine the shopping frequency of customers to identify who shops most frequently (find out the highest distinct count of `transactionNo` on customer level, be careful with those transactions that is not for shopping – `filter those transaction quantity <= 0`). Also, find out what products (column `productName_process`) ‘`this customer`’ typically buys based on the `Quantity` of products purchased.

Question 1.6

As the data scientist, you would like to build a basket-level analysis on the product customer buying (filter the ‘`df`’ dataframe `with df['Quantity']>0`). In this task, you need to:

1.6.1 Group by the `transactionNo` and aggregate the category of product (column `product_category`) into list on `transactionNo` level. Similarly, group and aggregate name of product (column `productName_process`) into list on `transactionNo` level.

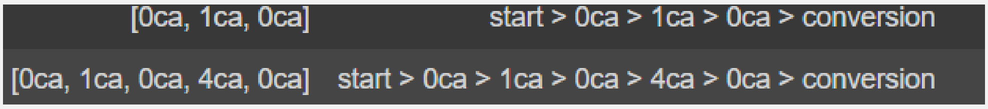
1.6.2 Removing duplicates on adjacent elements in the list from `product_category` you obtained from 1.6.1, such as `[product category 1, product category 1, product category 2, ...]` will be processed as `[product category 1, product category 2,....]`. After this processing, there will be no duplicates on on adjacent elements in the list. Please save your processed dataframe as ‘`df_1`’ and print the top 10 rows.

Question 1.7

Continue work on the results of question 1.6, now for each of the transaction, you will have a list of product categories. To further conduct the analysis, you need to finish below by using dataframe ‘`df_1`’:

1.7.1 Create new column `prod_len` to find out the length of the list from `product_category` on each transaction. Print the first five rows of dataframe ‘`df_1`’.

1.7.2 Transform the list in `product_category` from `[productcategory1, productcategory2...]` to ‘`start > productcategory1 > productcategory2 > ... > conversion`’ with new column `path`. You need to add ‘`start`’ as the first element, and ‘`conversion`’ as the last. Also you need to use ‘ `>` ’ to connect each of the transition on products (there is a space between the elements and the transition symbol `>`). The final format after the transition is given in example as below fig. 2. Define the function `data_processing` to achieve above with three arguments: `df` which is the dataframe name, `maxlength` with default value of 3 for filtering the dataframe with `prod_len" <=maxlength` and `minlength` with default value of 1 for filtering the dataframe with `prod_len >=minlength`. The function `data_processing` will return the new dataframe ‘`df_2`’. Run your defined function with dataframe ‘`df_1`’, `maxlength = 5` and `minlength = 2`, print the dataframe ‘`df_2`’ with top 10 rows.



[0ca, 1ca, 0ca]	start > 0ca > 1ca > 0ca > conversion
[0ca, 1ca, 0ca, 4ca, 0ca]	start > 0ca > 1ca > 0ca > 4ca > 0ca > conversion

Figure 2: Example of the transformation on 1.7.2, left column is before the transformation, right column is after the transformation. After transformation, it is not list anymore

Hint: you might consider to use `str.replace()` syntax from default python 3.

Question 1.8

Continue to work on the results of question 1.7, the dataframe '`df_2`', we would like to build the `transition matrix` together, but before we actually conduct the programming, we will need to finish few questions for exploration:

1.8.1 Check on your transaction level basket with results from question 1.7, could you please find out respectively how many transactions ended with pattern '`... > 0ca > conversion`' / '`... > 1ca > conversion`' / '`... > 2ca > conversion`' / '`... > 3ca > conversion`' / '`... > 4ca > conversion`' (1 result for each pattern, total 5 results are expected).

1.8.2 Check on your transaction level basket with results from question 1.7, could you please find out respectively how many times the transactions contains '`0ca > 0ca`' / '`0ca > 1ca`' / '`0ca > 2ca`' / '`0ca > 3ca`' / '`0ca > 4ca`' / '`0ca > conversion`' in the whole data (1 result for each pattern, total 6 results are expected and each transaction could contain those patterns multiple times, such as '`start > 0ca > 1ca > 0ca > 1ca > conversion`' will count '`two`' times with pattern '`0ca > 1ca`', if there is not any, then return 0, you need to sum the counts from each transaction to return the final value).

1.8.3 Check on your transaction level basket with results from task question 1.7, could you please find out how many times the transactions contains '`...> 0ca > ...`' in the whole data (1 result is expected and each transaction could contain the pattern multiple times, such as '`start > 0ca > 1ca > 0ca > 1ca > conversion`' will count '`two`' times, you need to sum the counts from each transaction to return the final value).

1.8.4 Use the 6 results from 1.8.2 to divide the result from 1.8.3 and then sum all of them and return the value.

Hint: you might consider to use `endswith` and `count` functions from default python 3.

Question 1.9

Let's now look at the question 1.6 again, you have the list of product and list of product category for each transaction. We will use the `transactionNo` and `productName_process` to conduct the Association rule learning.

1.9.1 Work on the dataframe `df` from question 1.2 (filter out the transaction with negative quantity value and also only keep those top 100 products by ranking the sum of quantity) and build the transaction level product dataframe (each row represents `transactionNo` and `productName_process` become the columns, the value in the column is the `Quantity`).
Hint: you might consider to use `pivot` function in pandas.

1.9.2 Run the apriori algorithm to identify items with minimum support of 1.5% (only looking at baskets with 4 or more items).
Hint: you might consider to use `mlxtend.frequent_patterns` to run apriori rules.

1.9.3 Run the apriori algorithm to find the items with `support >= 1.0%` and `lift > 10`.

1.9.4 Please explore three more examples with different support / confidence / lift measurements (you could leverage your rule mining with one of the three measurements or all of them) to find out any of the interesting patterns from the Association rule learning. Save your code and results in a clean and tidy format and writing down your insights.

Question 1.10

After we finished the Association rule learning, it is a time for us to consider to do customer analysis based on their shopping behaviours.

1.10.1 Work on the dataframe `df` from question 1.2 and build the customer product dataframe (each row represents single `customerNo` and `productName_process` become as the columns, the value in the columns is the aggregated `Quantity` value from all transactions and the result is a N by M matrix where N is the number of distinct `customerNo` and M is the number of distinct `productName_process`. Please filter out the transaction with negative quantity value and also only keep those top 100 product by ranking the sum of quantity).

1.10.2 Use the customer-product dataframe, let's calculate the Pairwise Euclidean distance on customer level (you will need to use the product Quantity information on each customer to calculate the Euclidean distance for all other customers and the result is a N by N matrix where N is the number of distinct `customerNo`).

1.10.3 Use the customer Pairwise Euclidean distance to find out the top 3 most similar customer to `CustomerNo == 13069` and `CustomerNo == 17490`.

1.10.4 For the customer `CustomerNo == 13069`, you could see there are some products that this customer has never shopped before, could you please give some suggestions on how to recommend these product to this customer? please write down your suggestions and provide a coding logic (steps on how to achieve, not actual code).

Part II

Sales Prediction

There are **3** questions in this part, totaling **40** marks. Each question is worth **10** marks. Additionally, the quality of your explanation in both the report and video will collectively be worth **10** marks.

You are required to use `Google Colab` to finish all the coding in the *code block cell*, and provide sufficient coding comments, and also save the result of running as well.

In this part, we will focus only on two columns `revenue` with `transaction_date` to form the revenue time series based on `transaction_date`. We will use the dataframe `df` from question 1.2 (without any filtering on transactions) to finish below sub-tasks:

Question 2.1

You are required to explore the revenue time series. There are some days not available in the revenue time series such as `2019-01-01`. Please add those days into the revenue time series with default revenue value with the mean value of the revenue in the whole data (without any filtering on transactions). After that, decompose the revenue time series with addictive mode and analyses on the results to find if there is any seasonality pattern (you could leverage the `M05A` material from lab session with default setting in `seasonal_decompose` function).

Question 2.2

We will try to use time series model `ARIMA` for forecasting the future. you need to find the best model with different parameters on `ARIMA` model. The parameter range for p,d,q are all from `[0, 1, 2]`. In total, you need to find out the best model with lowest `Mean Absolute Error` from 27 choices based on the time from "Jan-01-2019" to "Nov-01-2019" (you might need to split the time series to train and test with grid search according to the `M05B` material).

Question 2.3

There are many deep learning time series forecasting methods, could you please explore those methods and write down the necessary data wrangling and modeling steps (steps on how to achieve, not actual code). Also please give the reference of the deep learning time series forecasting models you are using.