CMPT459 D100 Fall 2022
Data Mining
Instructor: Martin Ester
TAs: Atia Hamidizadeh, Arash Khoeini

Course Project Milestone 1

Deadline: Friday, October 14th, 2022

Total Marks: 100

Introduction

In our course project, you will work with publicly available COVID-19 datasets.

- Case dataset for training (*cases_2021_train.csv*)
  - This file contains the training data for individual cases, i.e., cases that tested positive for COVID-19.
- Case dataset for testing (*cases_2021_test.csv*)
  - This file contains the testing data for individual cases (similar to the training data), but without the ‘outcome’ label.
- Location dataset (*location_2021.csv*)
  - This file contains the number of cases for each location.

These files were obtained from the following open-source repositories:

- https://github.com/beoutbreakprepared/nCoV2019
- https://github.com/CSSEGISandData/COVID-19

The datasets have been filtered for use in our project. To ensure consistency, everyone will use the March 31st, 2021 version of the datasets, which can be downloaded from:
https://github.com/shumanpng/CMPT459-D100-SPRING2022/tree/main/dataset

The data mining task will be to predict the ***outcome group*** (i.e., hospitalized, nonhospitalized, deceased, recovered) of a case. In this first milestone of the course project, you will preprocess the data as specified later in this document. In the next milestone, you will build models and use those models to predict the outcome groups on the *cases_2021_test.csv* dataset. The performance of the models will be evaluated based on the predictions on the test set, and hence it is important to treat *cases_2021_test.csv* and *cases_2021_train.csv* similarly.

Getting Started

0.1 Set up a conda environment with Python version 3.7

```
conda create -n cmpt459 python=3.7
```

For guidance on how to manage conda environments, check out this documentation: https://docs.conda.io/projects/conda/en/latest/user-guide/tasks/manage-environments.html

Packages that you may need are:

- **Pandas**
  - Command to install: `conda install pandas`
- **Numpy**
  - Command to install: `conda install numpy`
- **Jupyter notebook**:
  - Command to install: `conda install jupyter`
- **Seaborn**:
  - Command to install: `conda install seaborn`
- **Matplotlib**:
  - Command to install: `conda install -c conda-forge matplotlib`

0.2 Style guide

Follow the **PEP 8 style guide** for your Python code. If the TA cannot easily read/understand your code and have trouble reproducing your results, you will lose marks on your project milestone 1.

Preprocessing Tasks

1.1 Cleaning messy outcome labels (9 marks)

Datasets involved: *cases_2021_train.csv*

Load the *cases_2021_train.csv* data file using the following line of Python code:

```
cases_train = pd.read_csv('../data/cases_2021_train.csv')
```

You will find the **outcome_group** column to be filled with NaN values. The **outcome_group** values will be the labels that you will be predicting. You will need to clean up the raw outcome values in the **outcome** column of the loaded *cases_train* data frame. You can view all of the possible **outcome** values (and their frequencies) using this:

```
cases_train.groupby('outcome').size()
```

Real-world data is *messy*. For the same “outcome” (e.g., deceased), you will see 6 different ways of expressing it, namely “Dead”, “Death”, “Deceased”, “Died”, “death”, and “died.” Your task is to clean and summarize these semantically similar outcomes into four distinct **outcome_group** values (i.e., hospitalized, nonhospitalized, deceased, recovered). Outcomes that belong to each distinct **outcome_group** are listed in the table below:

outcome_group	**corresponding outcomes (*outcome* column)**
hospitalized	‘Discharged’, ‘Discharged from hospital’, ‘Hospitalized’, ‘critical condition’, ‘discharge’, ‘discharged’
nonhospitalized	‘Alive’, ‘Receiving Treatment’, ‘Stable’, ‘Under treatment’, ‘recovering at home 03.03.2020’, ‘released from quarantine’, ‘stable’, ‘stable condition’
deceased	‘Dead’, ‘Death’, ‘Deceased’, ‘Died’, ‘death’, ‘died’
recovered	‘Recovered’, ‘recovered’

You will fill in ‘recovered’ in the **outcome_group** column for rows (cases/samples) in the *cases_train* data frame that you previously loaded if the value in their **outcome** column is either ‘Recovered’ or ‘recovered’. Repeat the same procedure for all rows in the *cases_train* data frame and fill values into the **outcome_group** column according to the table above. Report the number of cases for each **outcome_group** label.

After you fill in values for the **outcome_group** column, *remove* the **outcome** column from *cases_2021_train.csv*.

1.2 Outcome labels (1 mark)

Datasets involved: *cases_2021_train.csv, cases_2021_test.csv*

What type of data mining task is the prediction of the outcome_group labels in the *cases_2021_train.csv* and *cases_2021_test.csv* datasets?

1.3 Exploratory Data Analysis (EDA) (20 marks)

Datasets involved: *cases_2021_train.csv, cases_2021_test.csv, location_2021.csv*

Perform exploratory data analysis to gain an understanding of the datasets. For all three datasets, show visualizations and statistics for all attributes that make sense. For example, it would make sense to visualize and compute statistics for categorical and numerical attributes, whereas it would not make sense to do so for textual attributes such as the description. For some attributes (e.g., longitude and latitude), it may make sense to visualize the combination of two attributes. Finally, for every attribute print the number of missing values (NaN).

1.4 Data cleaning and imputing missing values (5 + 15 marks)

Datasets involved: *cases_2021_train.csv, cases_2021_test.csv, location_2021.csv*

Perform data cleaning steps, mainly on the age column. Remove all of the entries (cases) with missing age values. Then, reduce different formats (e.g., 20-29, 25-, 13 months, 0.3), to a standard integer format (e.g., 25, 0, 1). Discuss how you reduced the different formats and justify your approach. Note that normally we will impute missing values, but we will not do so for the 'age' attribute because the ratio of missing values to non-missing values is too high.

For all the other attributes with missing values, discuss why and how (if applicable) you impute missing values.
Apply your imputation strategy to your datasets (*cases_2021_train.csv, cases_2021_test.csv, location_2021.csv*).

1.5 Dealing with outliers (15 marks)

Datasets involved: *cases_2021_train.csv, cases_2021_test.csv, location_2021.csv*

Which attributes have outliers? How did you find the outliers? How did you deal with them? Apply your strategy of dealing with outliers to all three datasets.

1.6 Joining the cases and location dataset (20 marks)

Datasets involved: *cases_2021_train.csv, cases_2021_test.csv, location_2021.csv*

The two datasets can be joined using some shared features. You can use 'province, country'. Present your strategy for joining the datasets and motivate your design decisions. Apply your join strategy to create a dataset of cases with additional features inherited from their locations. Report the number of rows (cases/samples) in the joined *cases train* and *cases test* datasets.

Hint: you may want to take a special look at how the countries 'South Korea' and 'United States' are expressed in the '**Country_Region**' column of the *location_2021.csv* dataset and modify them so that they are compatible with the country names listed in the '**country**' column of the *cases_2021_train.csv* and *cases_2021_test.csv* datasets.

1.7 Feature selection (15 marks)

Based on your response for task **1.2**, select as features a "relevant" subset of all attributes for the joined train and test cases datasets obtained from task **1.6**. Which attributes are selected and which attributes are discarded? Justify your approach.

Submission (Code + Report)

There are **two deliverables** for this milestone:

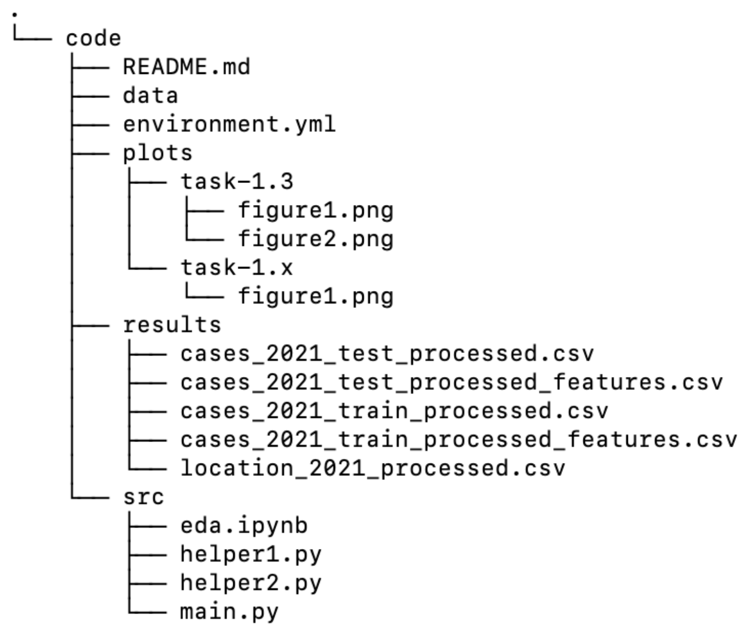
1. code.zip
2. report.pdf

Code

When you are ready to submit your code, please export your environment into an *environment.yml* file:

1. Activate the environment that you want to export: `conda activate cmpt459`
2. Export the environment: `conda env export > environment.yml`

Submit a ‘code.zip’ file with the following contents. It should contain the code, results, and figures obtained. The structure should look like this:



```
.
└── code
    ├── README.md
    ├── data
    ├── environment.yml
    ├── plots
    │   ├── task-1.3
    │   │   ├── figure1.png
    │   │   └── figure2.png
    │   └── task-1.x
    │       └── figure1.png
    ├── results
    │   ├── cases_2021_test_processed.csv
    │   ├── cases_2021_test_processed_features.csv
    │   ├── cases_2021_train_processed.csv
    │   ├── cases_2021_train_processed_features.csv
    │   └── location_2021_processed.csv
    └── src
        ├── eda.ipynb
        ├── helper1.py
        ├── helper2.py
        └── main.py
```

- *`data/`* folder for the original datasets (where the three input csv files are stored)
  - **Note:** Do **NOT** include the three original input csv files in the submission. Retain only the ‘data/’ folder structure as shown in the image above.
- *`plots/`* for the figures obtained
  - *`plots/task-1.3/`* ⇒ should contain all the figures that are relevant to task 1.3
  - *`plots/task-1.x/`* ⇒ should contain all the figures that are relevant to task 1.x (x=4,5,6,7). Create these task-specific directories as you feel appropriate.
- *`results/`* ⇒ for the results obtained after processing
  - *`results/cases_2021_test_processed.csv`* ⇒ should be the *cleaned* and *joined* dataset from task 1.6

    - `results/cases_2021_train_processed.csv` ⇒ should be the *cleaned* and *joined* dataset from task 1.6
    - `results/cases_2021_test_processed_features.csv` ⇒ should be the selected features from task 1.7
    - `results/cases_2021_train_processed_features.csv` ⇒ should be the selected features from task 1.7
    - `results/location_2021_processed.csv` ⇒ should be the cleaned/processed dataset from task 1.6 (this dataset should *not* be joined with other datasets)
- `src/` for code and scripts
    - `src/eda.ipynb` ⇒ for task 1.3 (EDA)
    - `src/main.py` ⇒ for tasks 1.1, 1.4, 1.5, 1.6, 1.7
    - `src/helper-n.py` ⇒ for any helper functions that you want to use in main.py. You may add more helper function scripts if you wish.
- `README.md` ⇒ for any special instructions for code execution (e.g., if 3rd party APIs are used, high running time, etc.)
- `environment.yml` ⇒ info for the environment that you used (this would help the TA reproduce your results)

Running '`src/eda.ipynb → Cell → Run All`' and '`python src/main.py`' should reproduce all the reported results with the same directory structure as described above. You are free (and encouraged) to split codes into helper functions and use them in `main.py`.

Make sure to use **relative paths** (e.g., '`../data/location_2021.csv`') to access files and folders within the code. **Do not use absolute paths** (e.g., '`Users/shumanp/Desktop/CMPT459/code/data/location_2021.csv`').

Report

Briefly explain the approaches and steps taken in the preprocessing tasks and answer the question asked in task 1.2. The report should **NOT** be more than 2 pages. Submit a 'report.pdf' file.