Project Handover Document

City of Greater Geelong

Data2Intelligence

Trimester 3, 2021

Table of Contents

1. Project Information

1.1. Client/Product Owner

Mr. Rizwan Haque
rhaque@geelongcity.vic.gov.au

1.2. Academic Mentor/Supervisor

Fatemeh Ansarizadeh
Casual Academic, School of Information Technology
f.ansarizadeh@deakin.edu.au

1.3. Project Team

Geelong City
Data2Intelligence Consulting

Student ID	Student Name	Role
220384895	**RAO SIDDHANT YADAV**	Machine learning senior
220500496	**RAKSHITH RACHENAHALLI BETTEGOWDA**	Web development senior
	KAIYUAN JIANG	Machine Learning senior
	SAHIT MANTRIPRAGADA	Data visualization senior
221045868	**RAHUL KUMAR**	Machine learning junior
219572977	**ABDULLAH ABDULAZIZ M ALGAMDI**	Web development junior
220034661	**SALEH MOHAMMED M ALHARBI**	Web development junior
217429086	**ANYU XU**	Data visualization junior
221153273	**ANUGRA SARA THOMAS**	Machine Learning and Data Setup junior
216327327	**CHRIS CHEN**	Data Visualization junior

2. Project Overview

In 2015 Geelong City launched their Digital Geelong Strategy which aimed to guide the city efforts in becoming a clever and creative city region. The aim of this initiative is to have Geelong be internationally recognised as 'a clever and creative city-region that is forward looking, enterprising and adaptive, and cares for its people and environment' by 2047.

Since the inception of this strategy Geelong City has fostered several innovative projects which incorporate improvements to free public Wi-Fi, smart on-street parking, smart waste management, and the launch of the 'Geelong Data Exchange repository'. More information on the background of this drive and explicit activities & projects can be located on the Geelong Smart City website. (https://www.geelongaustralia.com.au/smartcity/default.aspx)

This project team works with a representative from Geelong City Council to build and develop smart city initiatives, functioning as a melting pot for ideas and their rapid development.

In previous trimesters, the project team has used machine learning algorithm to analyze and predict trends of Tree and Pedestrian problem, also generated dashboard as resources showed on website**. In this trimester (T3 2021), the team has contributed on following components:**

1. Developed existing datasets and machine learning (ML) models for Tree & Pedestrian Problem.
2. Added 2 new projects (Wi-Fi & Fire Problem).
3. Added new features to Pedestrian problem- 'Reverse Geocoding & creating lockdown stages'.
4. Added features to Wi-Fi problem- 'Clustering & Predictive modelling (Boosting & other supervised machine learning models), 'Neural Networks (LSTM)'.
5. Done LSTM on Fire Problem too.
6. Redesigned & created new dashboards from scratch to cover more & essential information, answer more problem to further fit requirement.
7. Further optimised the website by adding enhanced UI | Homepage and Inner pages, provided utilities like 'subscription' and 'contact us', added two factor authentication to improve security, integrated analysis from data visualization team, deployed Machine learning models & hosted the website on 'Heroku'.

During ML development, we mined more data to improve existing dataset, added more ML models, also adjusted existing ML model and later selected the best model for our projects. For reasons of GPU function, the platform we chose to test models on Machine Learning front is Google Colab Pro. Base on outcomes from ML and case study, we continued our data visualisation development on Tableau. Where we redesigned dashboards for tree problem and pedestrian problem, also created map plots for fire problem & Wi-Fi problem. And our website was delivered using Django web framework & has been configured for deployment on a live server (GCP, Heroku).

3. User Manual

In order to perform the front-end and back-end web development work, following software are required on system:

1. **Python**
   Because Django is a Python web framework, it is necessary to have Python version 3.8.5 installed on the computer. It is possible to obtain the needed Python version from https://python.org/downloads/ location.

   Install Python and then open a command prompt and check that the Python version shown corresponds to the version you installed by running the following command. It is possible to use the Python language with this command: **python —version**

```
c:\D2I-Geelong>python --version
Python 3.8.5
```

2. **Setting up a virtual environment**
   It is essential that each Django project be executed in its own environment.
   To create a virtual environment, open a new command prompt and navigate to the folder containing the D2I-Geelong project. It's necessary to issue the following command:
   - **python -m venv project-name**
   - **env\Scripts\activate**

```
c:\D2I-Geelong>python -m venv env

c:\D2I-Geelong>env\Scripts\activate

(env) c:\D2I-Geelong>
```

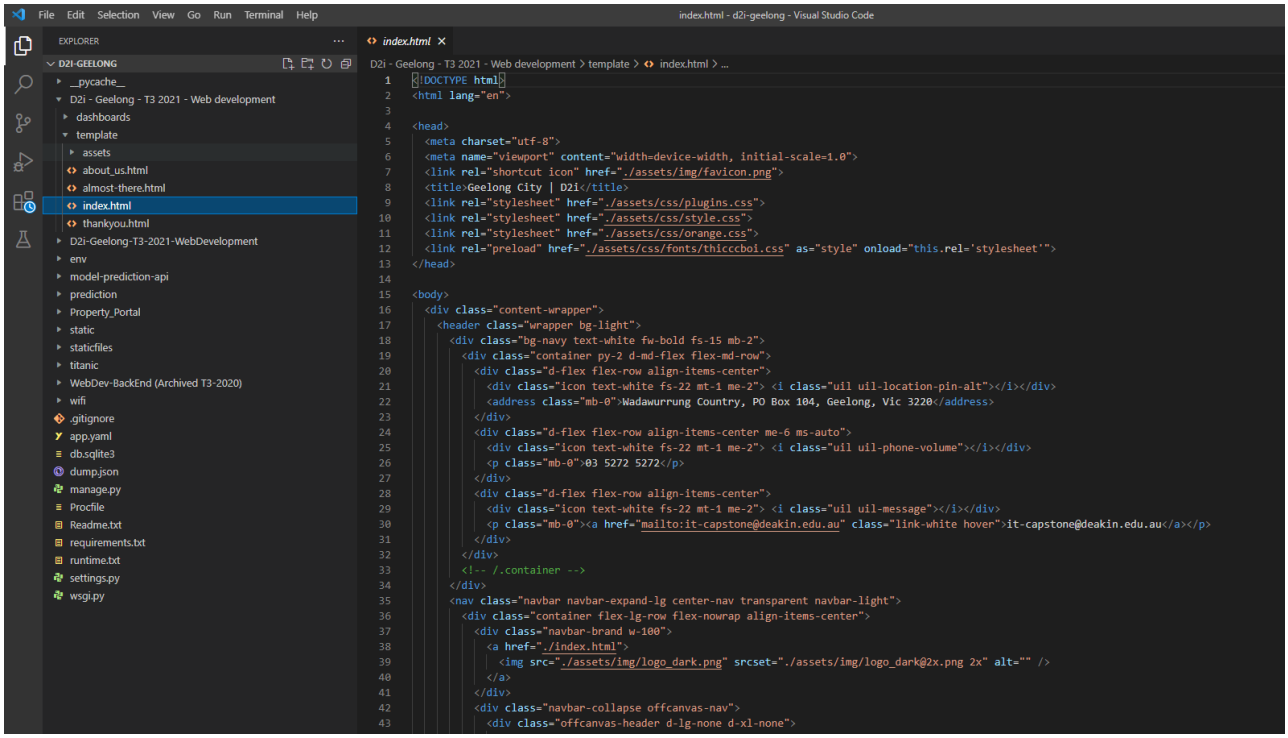
   The command prompt will display "(project-name)" when the virtual environment is active. You'll have to re-activate the environment each time you open a new command prompt.

3. **Install Dependencies/Requirements**
   All the dependencies and requirements are listed in requirements.txt. Use following command to install the dependencies/requirement. It will take some time to download and install all of the necessary components.
   - **pip install -r requirements.txt**

```
(env) c:\D2I-Geelong>pip install -r requirements.txt
Collecting asgiref==3.2.10
  Using cached asgiref-3.2.10-py3-none-any.whl (19 kB)
Collecting cachetools==4.2.2
  Using cached cachetools-4.2.2-py3-none-any.whl (11 kB)
Collecting certifi==2020.6.20
  Using cached certifi-2020.6.20-py2.py3-none-any.whl (156 kB)
```

4. **Install Visual Studio Code**
   Download Visual Studio Code from https://code.visualstudio.com/ location. After download, install and run it.

   Drag the D2i-Geelong project folder into Visual Studio Code window.



4. Completed Deliverables

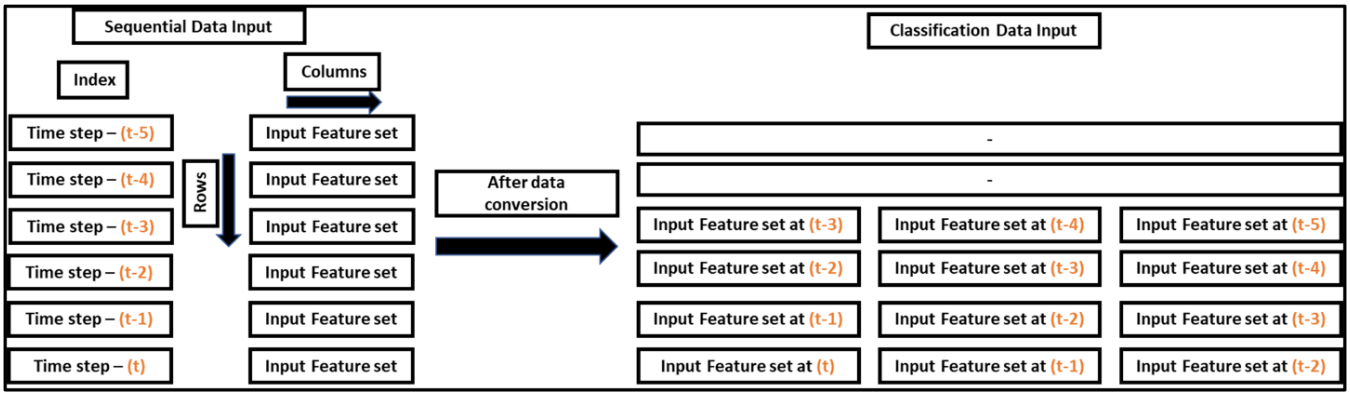
The team for this trimester was generally divided into three sub teams-

1. **Machine Learning**
2. **Data Visualization**
3. **Web Development**

The Machine Learning team was further divided into the Trees & Fire team and the Pedestrian & Wi-Fi team. The completed deliverables for each group are discussed next.

Series to supervised conversion (Fire & Wi-Fi Problem):

For a better understanding of this conversion, let us review the diagram shown below and understand. We must put dates (timestamps) as index and keep the feature set. Next, we keep these features as embeddings, so the last sample is a column concatenation for timestamp t, t-1 and t-2. Similarly, this follows for the third last sample and the last two samples are dropped from the entire dataset. In our code we have kept this to t and t-1 and removed the last sample. Please refer to the snippet below for better understanding.



Trees Problem

Idea:

We have worked on the concept of tree data analysis and tree prediction that had been proposed by former team members. The main goal of tree data analysis and tree prediction is to determine the various names of trees that can be planted in different regions of Geelong City. After that, we shared the idea with the client, who was quite enthusiastic about us pushing forward with it.

Data Source:

The dataset for the Trees problem was obtained from the Geelong Data Exchange website.

About Dataset:

After acquiring the dataset from the Geelong Data Exchange website, it was discovered that it contained 152696 observations split over 30 characteristics. The tree data provided information about each tree that had been planted in various locations. It included details such as the tree's crown width, height, and other characteristics. In the table below, all of the features are listed.

DATA SET NAME	Trees Data
DATA SIZE	45,090 KB
DATE OF RELEASE	15/08/2021
NO. OF FILES	1
NO. OF ATTRIBUTES	30
NO. OF DATA RECORDS	152,696
DATA SOURCE PROVIDER	https://www.geelongdataexchange.com.au/pages/home/
DATA PRIVACY	Publicly available to everyone

NOTES

Prepared by:	*Anugra Sara Thomas*
Point of Contact:	*Anugra Sara Thomas*, *asthoma@deakin.edu.au*
Team Members:	*Anugra Sara Thomas*, *Rao Siddhant Yadav*

Data Cleaning & pre-processing:

The dataset was discovered to have a significant number of missing values, which makes it extremely difficult to analyse. To handle them (and apply Machine Learning Model algorithms), we employed Imputer, Dropna, and Fillna techniques as part of the pre-processing and cleaning procedure.

For handling and pre-processing the features, the following steps were taken:

i) We reserved over 5000 entries more.
ii) Dropped irrelevant columns such as tree_id, Category_txt, ule, compkey etc.
iii) Used the Simple Imputer, Dropna, and Fillna strategies to deal with null values.
iv) Plant width was presented as a range in the original data, and the lower and upper limits of the range were separated into two columns.
v) Similarly, the original data for dbh (diameter at breast height) was supplied as a range, which was split into two columns due to the bottom and upper limits of the range.
vi) The cleaned data contained 19 features after all of the data pre-processing and handling steps.

After all the cleaning, it was time to start some scaling on the data and hence a standard scaler was used on all the numeric attributes to scale the observations between -1 and 1. Also, before modelling, suburbs were categorically coded i.e., numbered starting from 1.

Data Modelling:

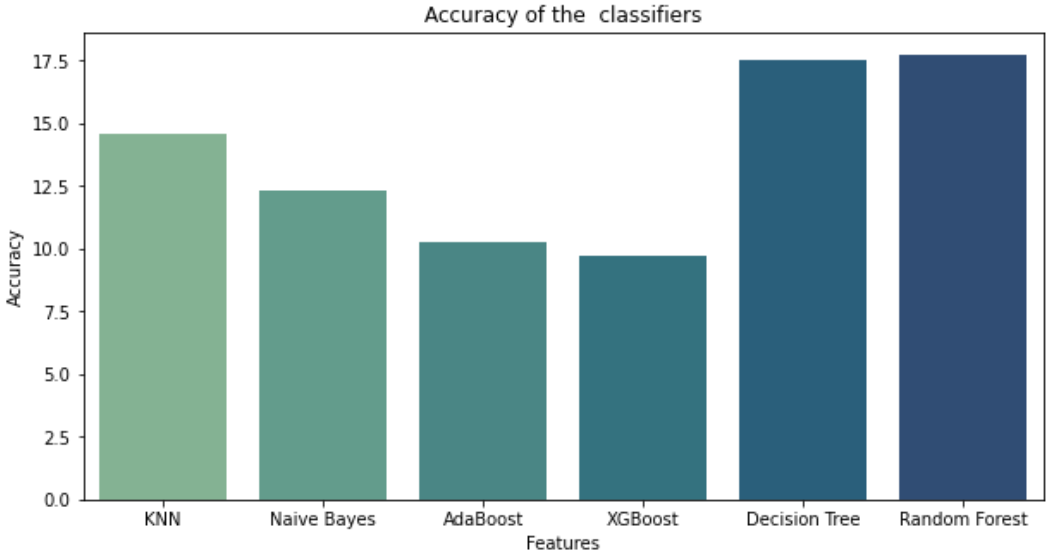
We performed hyperparameter tuning for all algorithms and also faced runtime issues. To determine the performance of the model, Accuracy of the model was included. In addition, the team also looked at weighted average f1-score which helped in determining the performance of the model. The main challenge that was to be dealt while modelling was, there were a lot of trees which had only one observation, which might cause the Machine Learning Models to perform poorly. And the team chose to disregard/ignore trees with fewer than five occurrences in the data.

To train and validate the model's performance, the dataset was divided into two parts: 60% for training and 40% for testing and validating. The following models were developed and evaluated by the team:

i. K-Nearest Neighbours
ii. Logistic Regression
iii. Naïve Bayes

iv. Decision Tree
v. Random Forest
vi. Support Vector Machine
vii. AdaBoost
viii. XGBoost

The final Accuracy and F1 score results enhanced, and we also incorporated run-time results for our model selection. On the basis of accuracy and F1 score results, Random Forest, K-nearest neighbours, and Decision tree were the best models. However, we have decided that the Decision tree as the best model because it takes the least computational time. Subsequently, the results we achieved were approximately 2% better than those previously obtained. Lastly, the Decision Tree model was saved and exported as 'pkl' file so that it can be deployed on the website.



Building the ML model for predicting name of the tree

The best model came out to be a Decision tree (based on accuracy score) having 'gini' criterion, max_depth=10.

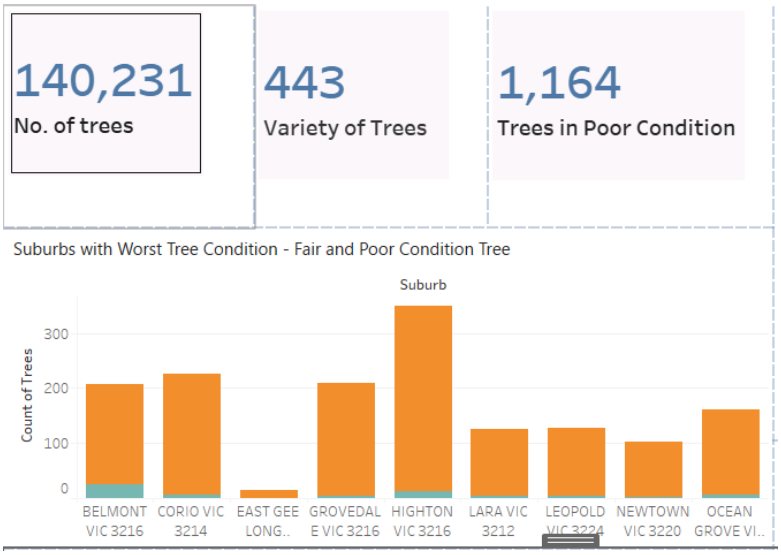
Analysis on the performance of best model

According to the above-mentioned graph, the model was functioning at a standard moderate level, with an accuracy score of only 17.5. Furthermore, the weighted average of the f1-score was 12.1 points, which is not particularly high. Accuracy was used to compare the models since it can be used as a scale to quantify how well the model performed and it is independent of the number of observations. The model's performance improves as the accuracy score rises (Depending on the model, in this scenario one has look at the weighted average precision and f1-score to avoid the under-fitting and over-fitting of the model).

Visualisations of Trees Problem (using Tableau)

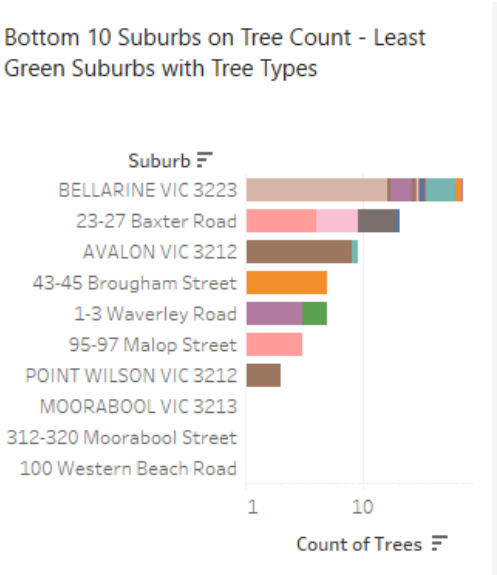
For visualisation purposes we chose Tableau. Tableau helped in visualising the data and provided some useful insights about the data. It helped in forming an interactive dashboard for the trees feature which the audience of the website would be viewing the same. Some of the visualisations that were created in tableau are as follows:

Suburbs with Worst Tree Condition - Fair and Poor Condition Tree:



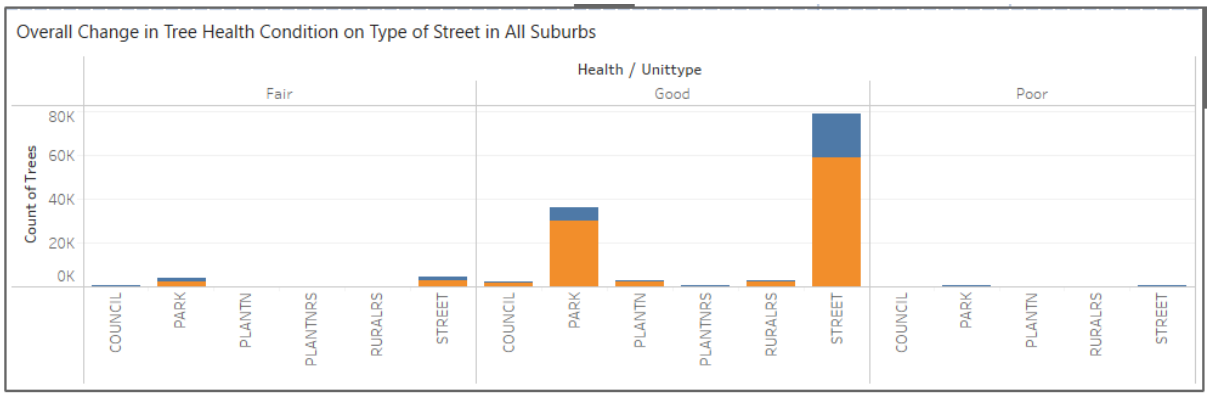
The figure of tree problem-1 shows the number of total count of trees collected in all suburb, and total categories of trees and total count of trees in the poor condition. And for the figure about the suburbs with worst tree condition, the orange colour means the structure of trees grown up fair, the green colour means the structure of trees grown up poor.

Bottom 10 Suburbs on Tree Count - Least Green Suburbs with Tree Types:

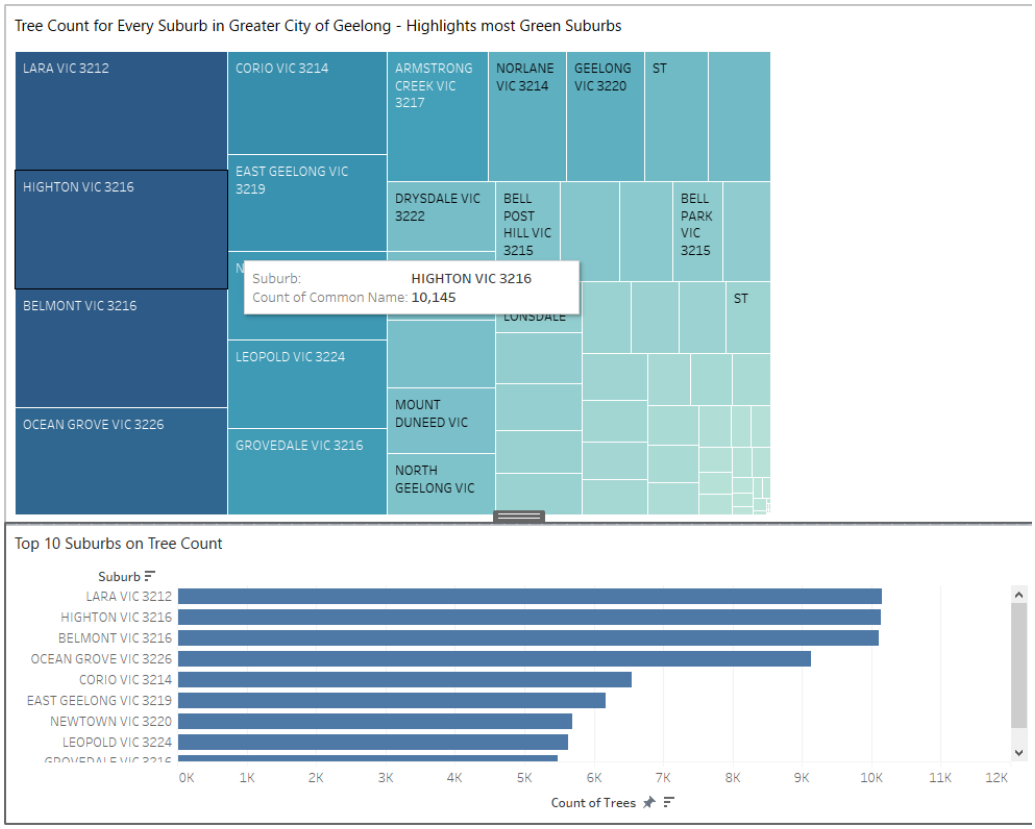


The visualization given above provides us with the 10 suburbs with the lowest number of trees, which means that the 10 suburbs with the least number of trees are located in Geelong. The suburbs with the least number of trees in Geelong are 100 Western Beach Road, MOORABOOL, 312-320 Moorabool Street, and only one tree is planted in these suburbs.

Overall Change in Tree Health Condition on Type of Street in All Suburbs:



According to the visualization chart given above, we can know the health of various street types of trees at different ages. We analyse the distribution of trees in various health states in different street types according to the visualization, as well as the proportion of different age states in health or other states. The orange colour means semi-mature, and the blue color means mature. We can easily find the street has the most tree count in good condition.



For the above figure, we use different types of charts to show tree count for every suburb, we use the ‘treemap’, which shows darker blue the colour is, the greater number of trees. And we also use the bars chart to show the Top 10 Suburbs on Tree Count.

Suggestions for the next squad:

- To get hold of the carbon footprint data and then link it with the trees data to analyse which trees reduce more carbon
- To work on improving the model that has been used here (as the model is performing moderately).
- To investigate modelling the Trees data using deep learning models (use of neural networks).

Fire Problem

Idea:

For the Fire prediction problem, we predict possibility of bushfires in and around the city using an external Fire data source merged with the weather data. It ensures that users or rather Geelong citizens subscribed to this feature, will get notified in case of fire Hazard based on their suburb so that they can reach safety accordingly. The idea is to include more hazards in this once we get more relevant data for different hazards in the future. This idea was pitched in by the team members

Data Source:

The dataset for the VIIRS fire and weather data for Fire problem were obtained from the NASA website.

About Dataset:

After acquiring the Fire dataset from the NASA website, it was discovered that it contained 8664911 observations split over 13 characteristics. It included details such as the latitude, longitude, brightness and other characteristics. In the table below, all of the features are listed.

DATA SET NAME	Fire
DATA SIZE	802067 KB
NO. OF FILES	1
NO. OF ATTRIBUTES	13
NO. OF DATA RECORDS	8664911
DATA SOURCE PROVIDER	https://firms2.modaps.eosdis.nasa.gov/country/
DATA PRIVACY	Publicly available to everyone

After acquiring the Weather dataset from the NASA website, it was discovered that it contained 1048575 observations split over 23 characteristics. It included details such as the 'MaxTemp, MinTemp, Date' and other characteristics. In the table below, all of the features are listed.

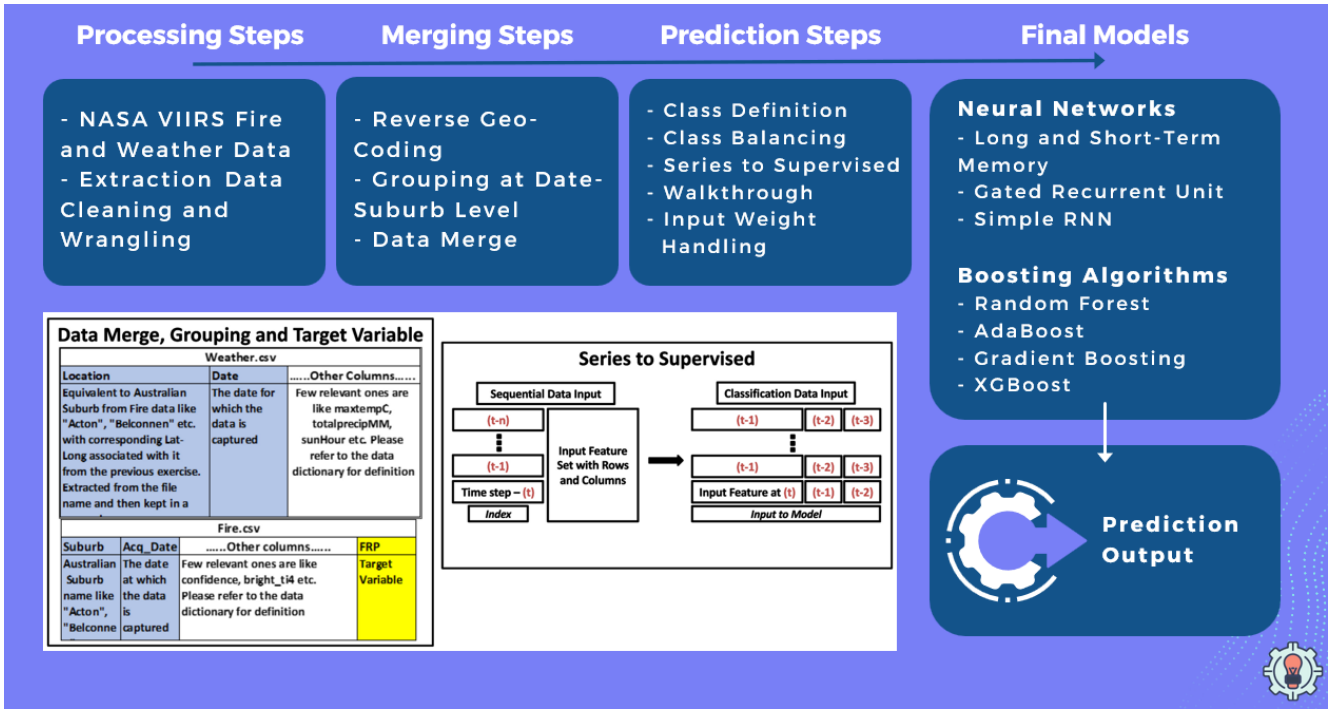
DATA SET NAME	Weather
DATA SIZE	348229 KB
NO. OF FILES	1
NO. OF ATTRIBUTES	23
NO. OF DATA RECORDS	1048575
DATA SOURCE PROVIDER	https://www.worldweatheronline.com/hwd/hfw.aspx
DATA PRIVACY	Publicly available to everyone

NOTES

Prepared by:	*Anugra Sara Thomas*
Point of Contact:	*Anugra Sara Thomas,* *asthoma@deakin.edu.au*
Team Members:	*Anugra Sara Thomas,* *Rao Siddhant Yadav*

Workflow

This is the detailed workflow of the following Fire problem:



In processing, we extracted both NASA VIIRS Fire data and weather data, and performed data cleaning and wrangling steps. We then performed reverse geocoding to map lat-long to respective suburbs and grouped this data to get weather-fire merged input for prediction exercise. We applied class definition and balancing first and then used series to supervised conversion to ensure we can use sequential data as input for a classification problem. We handled imbalance for Neural Network models by handling input weight and we see the list of Neural Network algorithms used by us including LSTM, GRU and Simple-RNN, followed by Boosting algorithms like Random forest, XGBoost etc. to give us our prediction output.

Step Involved:

The following steps were taken:

1. **Processing Steps**

   i) Fire Dataset
   - Filled the NaN values with 0
   - Dropped Irrelevant Columns such as acq_time, scan, track etc.
   - Reordered the Columns
   - Trimmed Down the dataset [using filter ()] for Reverse Geocoding

   ii) Weather Dataset
   - Dropped columns that is of no value to fire prediction (for e.g. , Snow, Moon, Temperature etc.)
   - Renamed columns

2. **Merging Steps**

   i. Fire Dataset
   - Reverse Geocoding (to convert the coordinates (latitude and longitude) to human-readable addresses (lat_suburb, long_suburb, suburb, region, state, country)
   - Filtered the dataset to only Victoria State
   - Filtering to match weather date range (2014-09-01 - 2020-11-30)
   - Grouping at Date-Suburb Level
   - Assigning 'nil' to any records where there is no council defined by state government

   ii. Weather Dataset
   - Reverse Geocoding (to convert the coordinates (latitude and longitude) to human-readable addresses (lat_suburb, long_suburb, suburb, region, state, country)
   - Filtered the dataset to only Victoria State
   - Grouping at Date-Suburb Level
   - Assigning 'nil' to any records where there is no council defined by state government

3. Merging Fire and Weather tables using Left Join

4. Removed redundant columns

5. Replaced 'acq_date' and 'suburb' NaN values with corresponding values from 'location' and 'date

6. Replaced frp NaN values with 0

7. Exported the weather_fire dataset to csv

8. **Prediction Steps**

   - Class Definition
   - Class Balancing

- Series to Supervised Conversion
- Walkthrough
- Input Weight Handling

9. Data Modelling:

We performed hyperparameter tuning for all algorithms and handled imbalance for Neural Network models by handling input weight. To determine the performance of the model, overall accuracy of the model was included. In addition, the team also looked at recall and F1 score which helped in determining the performance of the model. To train and validate the model's performance, the dataset was divided into two parts: 70% for training and 30% for testing and validating. The following ML models were developed and evaluated by the team:

1. **Boosting Algorithms**
    a. Random Forest
    b. Gradient Boosting
    c. AdaBoost
    d. XGBoost

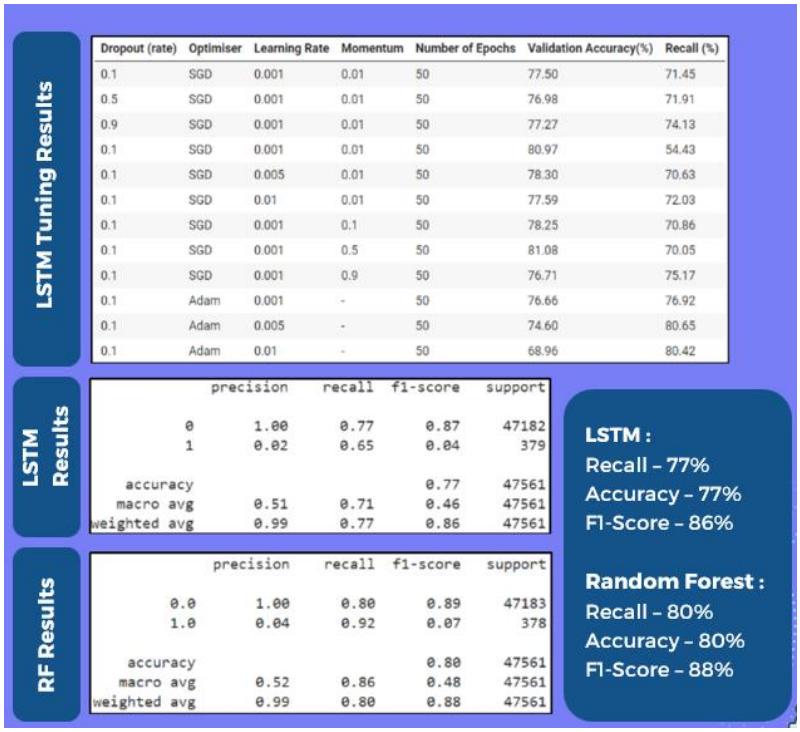
2. **Neural Networks**
    a. LSTM
    b. GRU
    c. SimpleRNN

On the basis of accuracy, recall and F1 score results, Random Forest and LSTM were the best models. However, we have decided that the Random Forest as the best model because it takes the least computational time. Lastly, the Random Forest model was saved and exported as pkl file so that it can be deployed on the website.

Random Forest:

```
RandomForestClassifier(max_depth=100, max_features='sqrt', min_samples_split=5,
                       n_estimators=400, random_state=42)
```

Results:



LSTM Tuning Results

Dropout (rate)	Optimiser	Learning Rate	Momentum	Number of Epochs	Validation Accuracy(%)	Recall (%)
0.1	SGD	0.001	0.01	50	77.50	71.45
0.5	SGD	0.001	0.01	50	76.98	71.91
0.9	SGD	0.001	0.01	50	77.27	74.13
0.1	SGD	0.001	0.01	50	80.97	54.43
0.1	SGD	0.005	0.01	50	78.30	70.63
0.1	SGD	0.01	0.01	50	77.59	72.03
0.1	SGD	0.001	0.1	50	78.25	70.86
0.1	SGD	0.001	0.5	50	81.08	70.05
0.1	SGD	0.001	0.9	50	76.71	75.17
0.1	Adam	0.001	-	50	76.66	76.92
0.1	Adam	0.005	-	50	74.60	80.65
0.1	Adam	0.01	-	50	68.96	80.42

LSTM Results

	precision	recall	f1-score	support
0	1.00	0.77	0.87	47182
1	0.02	0.65	0.04	379
accuracy			0.77	47561
macro avg	0.51	0.71	0.46	47561
weighted avg	0.99	0.77	0.86	47561

RF Results

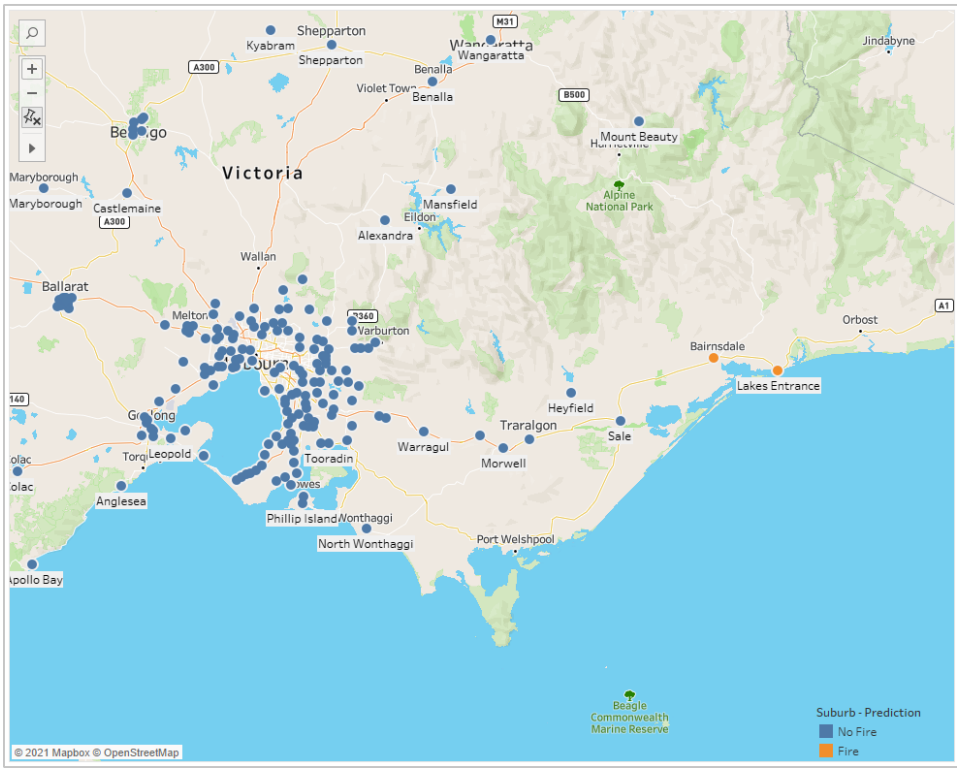
	precision	recall	f1-score	support
0.0	1.00	0.80	0.89	47183
1.0	0.04	0.92	0.07	378
accuracy			0.80	47561
macro avg	0.52	0.86	0.48	47561
weighted avg	0.99	0.80	0.88	47561

LSTM :
Recall – 77%
Accuracy – 77%
F1-Score – 86%

Random Forest :
Recall – 80%
Accuracy – 80%
F1-Score – 88%

We performed an hyperparameter tuning for all algorithms and we can look at the results for LSTM in the diagram above which was identified as the best Neural Network performer, but Overall, we have Random Forest identified as best model for Fire prediction at an overall recall and accuracy of 80% and more importantly a F1-Score of 88%. Latest results have been included in final product.

Visualization for Fire Data (using Tableau)



For this map, it clearly shows the distribution of fire for different regions in Victoria. The orange colour shows the region has fire disaster, and the blue colour shows the region does not has fire disaster.

Suggestions for the next squad:

In the iteration, we will continue to optimize the data prediction by the several following ways:
- Making data live or pipelining the live data to website.
- Making the data richer with new records.

Pedestrian & Wi-Fi Problem

Objective

The general idea is to monitor impact of citizen movement in Lockdown & Post-Lockdown stages & the places they have visited during the period in different regions. Through this data we predicted suburb-wise daily Wi-Fi usage by Geelong citizen. This can be used to plan better increase router device bandwidth or router count as per end user usage.

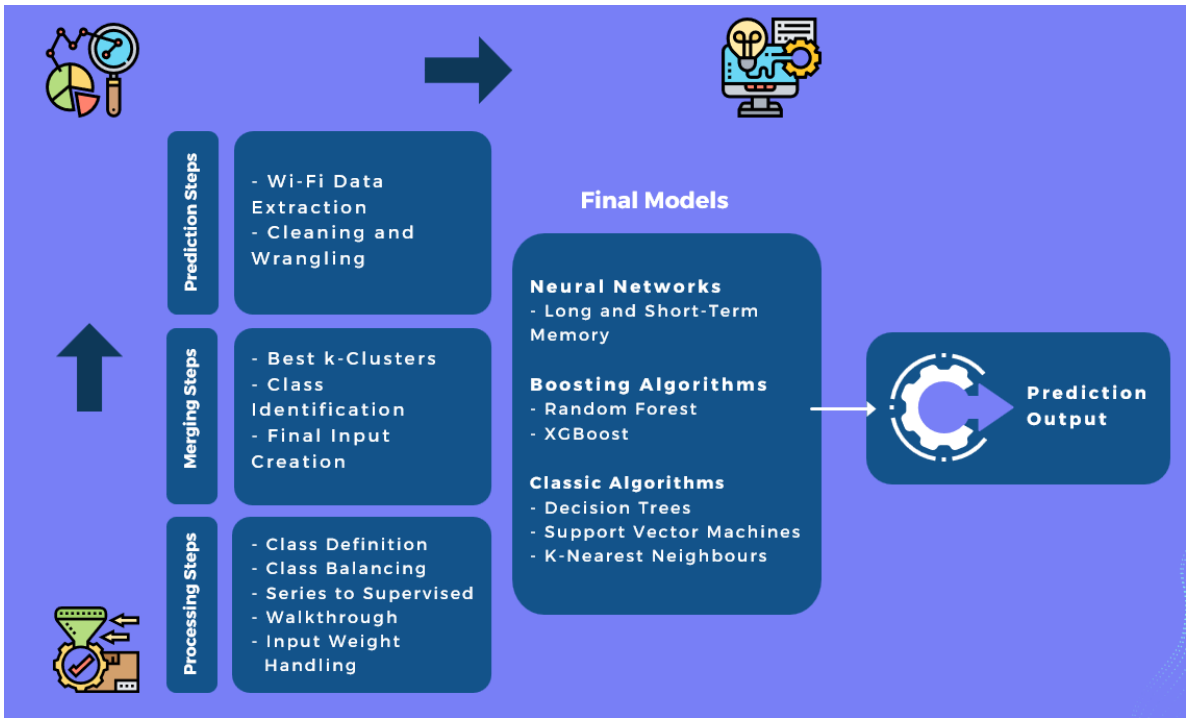
In Trimester 3 (T3) 2021, we have used the updated datasets and done data exploration, cleaning & merging of datasets then integrated 'Google Maps API' service in order to fetch location of Wi-Fi routers & building types (Hospitals, Libraries, Malls, Gyms & Restaurants). Later, we used 'Reverse Geocoding' to obtain address, post-codes & suburbs & also created 8 lockdown stages to monitor pedestrian movement. We also performed Geo-spatial & normal clustering to identify 'target class' & clustered the dataset based on Wi-Fi usage. Furthermore, we did Boosting & other supervised machine learning models using the clustered dataset & predicted the output for the last 2 days of our updated dataset using the best ML model. Apart from that, we also performed time series prediction using neural networks like LSTM, GRU & Simple RNN. At the end, we made visualizations to highlight foot count collected in different facilities & also made a Map plot to highlight future Wi-Fi predictions of last 2-days using the best ML model.

NOTES

Prepared by:	*Rahul Kumar*
Point of Contact:	*Rahul Kumar*, *krahul@deakin.edu.au*
Team Members:	*Rahul Kumar*, *Kaiyuan Jiang*

Workflow

This is the detailed workflow of the following Wi-Fi problem:



Data Extraction, Exploration, Cleaning & Merging

Data Extraction

First, we collected the dataset regarding the Pedestrian & Wi-Fi Problem, which is same & is available on 'www.geelongdataexchange.com.au' website for Geelong. We have used two datasets (historical & updated) that are available on website:

1. Wi-Fi Enabled Device Counts (from 2018 to Nov 2020)

2. Wi-Fi Enabled Device Counts (from Nov 2020 till 7th Dec 2021)

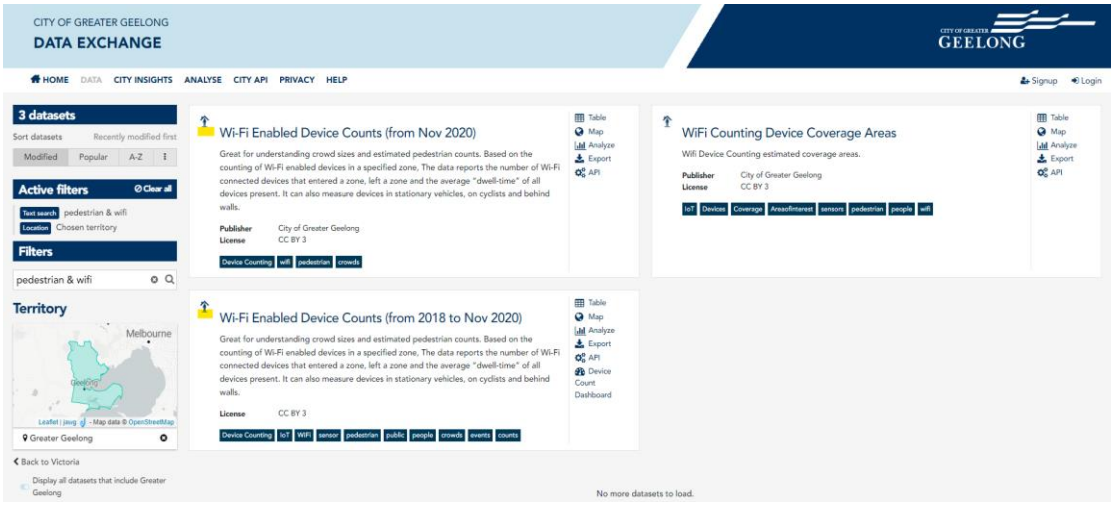


Figure 4.1 Datasets from the Geelong Data Exchange website

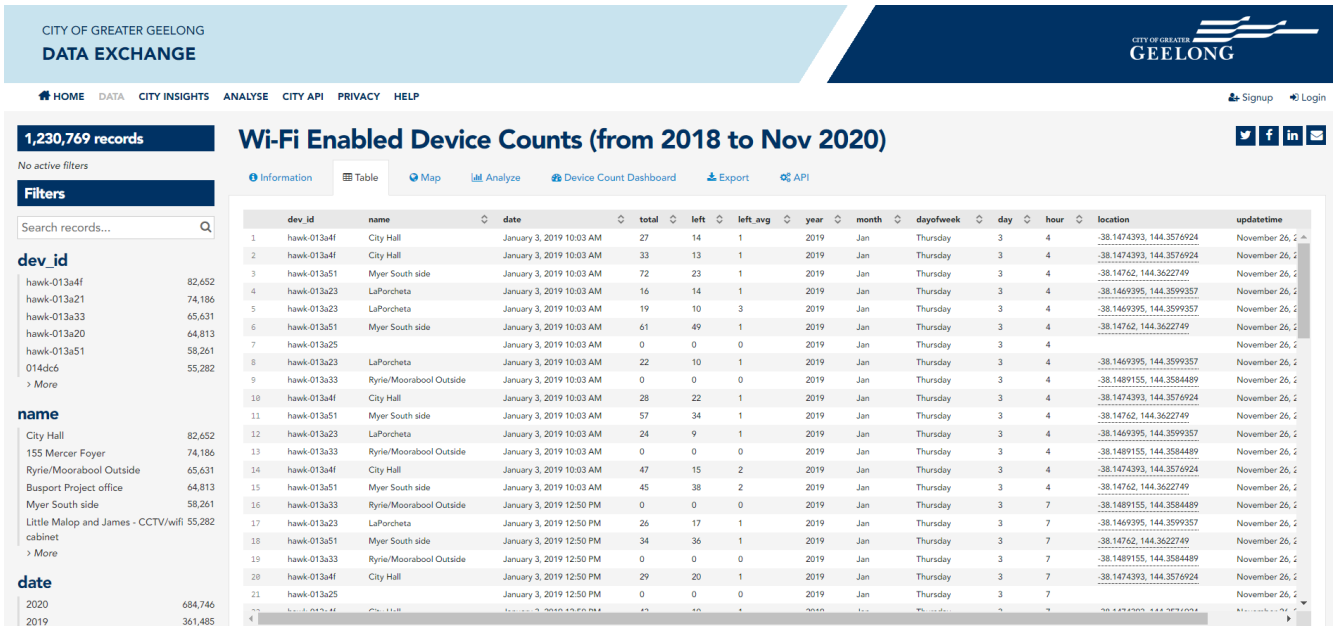


Figure 4.2 Raw dataset-1

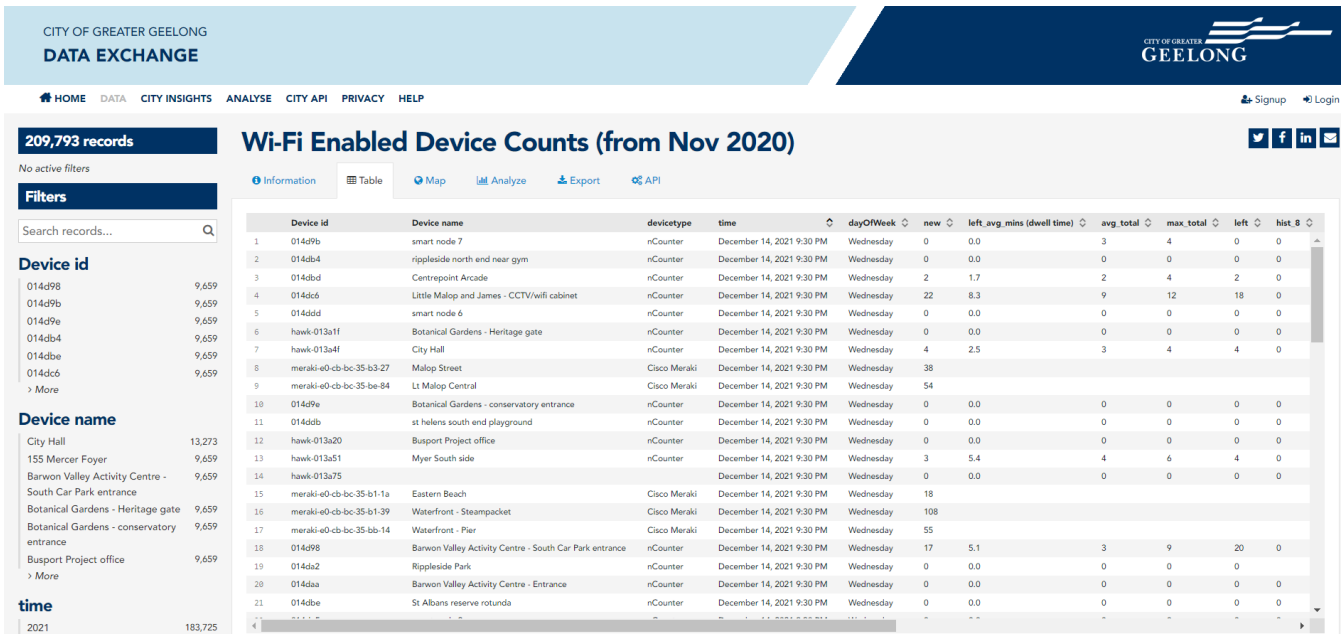


Figure 4.3 Raw dataset-2

Data Exploration & Cleaning (filtering anomalous data)

After collecting both datasets, we imported both the datasets & libraries required for data exploration and cleaning. We then printed out the dimension of the dataset, number of records the dataset contains, number of attributes in the dataset, dataset information & dataset description for exploring both the datasets.

Exploratory Data Analysis (EDA)

In [3]:
```
print("\nData size :",df.shape)
print("\nNumber of records :",len(df))
print("\nNumber of atrributes :",len(df.columns))
print("\nDataset Information :", df.info())
df.describe()
```

```
Data size : (1230769, 13)

Number of records : 1230769

Number of atrributes : 13
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1230769 entries, 0 to 1230768
Data columns (total 13 columns):
 #   Column      Non-Null Count    Dtype  
---  ------      --------------    -----  
 0   dev_id      1079034 non-null  object 
 1   name        964576 non-null   object 
 2   date        1055282 non-null  object 
 3   total       1230609 non-null  float64
 4   left        1230609 non-null  float64
 5   left_avg    1230609 non-null  float64
 6   year        1055282 non-null  float64
 7   month       1055282 non-null  object 
 8   dayofweek   1055282 non-null  object 
 9   day         1055282 non-null  float64
 10  hour        1055282 non-null  float64
 11  location    964576 non-null   object 
 12  updatetime  1230769 non-null  object 
dtypes: float64(6), object(7)
memory usage: 122.1+ MB

Dataset Information : None
```

Out[3]:

	total	left	left_avg	year	day	hour
count	1.230609e+06	1.230609e+06	1.230609e+06	1.055282e+06	1.055282e+06	1.055282e+06
mean	2.330043e+01	1.589944e+01	5.546636e-01	2.019640e+03	1.600553e+01	1.148710e+01
std	4.254929e+01	3.232956e+01	1.901706e+00	4.978064e-01	8.783271e+00	6.935308e+00
min	0.000000e+00	0.000000e+00	0.000000e+00	2.018000e+03	1.000000e+00	0.000000e+00
25%	1.000000e+00	0.000000e+00	0.000000e+00	2.019000e+03	8.000000e+00	5.000000e+00
50%	7.000000e+00	4.000000e+00	0.000000e+00	2.020000e+03	1.600000e+01	1.100000e+01
75%	2.600000e+01	1.600000e+01	1.000000e+00	2.020000e+03	2.400000e+01	1.800000e+01
max	4.810000e+02	4.150000e+02	2.550000e+02	2.020000e+03	3.100000e+01	2.300000e+01

After getting insights from the dataset, we moved on to data cleaning. In data cleaning, we first checked the total number of missing/null values in the dataframe. We also checked the missing/null values and their respective percentage of each column or feature from the dataset and later printed out the numerical columns.

In [4]:
```
# check if there are any missing values in the dataframe?
print(" \nCount total number of missing values in the DataFrame : \n\n", df.isnull().sum().sum())
```

```
Count total number of missing values in the DataFrame :

 1737523
```

In [5]:
```
# checking for null values

null_val = pd.DataFrame({'Null Values' : df.isna().sum(), 'Percentage Null Values' : (df.isna().sum()) / len(df) * (100)})
null_val
```

Out[5]:

	Null Values	Percentage Null Values
dev_id	151735	12.328471
name	266193	21.628185
date	175487	14.258321
total	160	0.013000
left	160	0.013000
left_avg	160	0.013000
year	175487	14.258321
month	175487	14.258321
dayofweek	175487	14.258321
day	175487	14.258321
hour	175487	14.258321
location	266193	21.628185
updatetime	0	0.000000

In [6]:
```
numerics = ['int16', 'int32', 'int64', 'float16', 'float32', 'float64']
numeric_df = df.select_dtypes(include=numerics)
print('Number of numerical columns are:', len(numeric_df.columns))
numeric_df
```

```
Number of numerical columns are: 6
```

Out[6]:

	total	left	left_avg	year	day	hour
0	27.0	14.0	1.0	2019.0	3.0	4.0
1	33.0	13.0	1.0	2019.0	3.0	4.0
2	72.0	23.0	1.0	2019.0	3.0	4.0
3	16.0	14.0	1.0	2019.0	3.0	4.0
4	19.0	10.0	3.0	2019.0	3.0	4.0
...	...	...	...	...	...	...
1230764	17.0	4.0	1.0	2020.0	8.0	3.0
1230765	26.0	6.0	1.0	2020.0	8.0	3.0
1230766	188.0	200.0	1.0	2020.0	8.0	2.0
1230767	242.0	226.0	1.0	2020.0	8.0	2.0
1230768	0.0	2.0	1.0	2020.0	8.0	2.0

1230769 rows × 6 columns

After checking all the missing/null values, for **data cleaning** we used 'Simple Imputer & fillna' method to retain as much records as possible & also done 'dropna' for cleaning or dropping irrelevant data.

After data cleaning, we have again checked the number of missing/null values & their respective percentages of each column. Later, printed out the cleaned data information and saved both the cleaned files to a CSV file.

After getting the cleaned records, we've **merged** both the cleaned datasets by taking common & relevant feature sets from both the datasets in order to use 'Google Maps API' service to fetch location of Wi-Fi routers, building types & monitor citizen movements across those buildings. Furthermore, saved the merged dataframe to a CSV file.

In [4]:
```
# displaying the columns of regularupdated_cleaned_wifi_data dataset
pd.set_option("display.max_columns", None)
df1.head(1)
```

Out[4]:

	Device id	Device name	time	dayOfWeek	avg_total	max_total	new	left_avg_mins (dwell time)	left	hist_8	hist_7	hist_6	hist_5	hist_4	hist_3	hist_2	hist
0	014d9e	Botanical Gardens - conservatory entrance	2021-08-25T06:30:00+05:30	Wednesday	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

In [5]:
```
# removing the columns form regularupdated_cleaned_wifi_data which are not present in cleaned_wifi_data dataset
# storing the results in new_df1
new_df1 = df1.drop(['devicetype','avg_total', 'new','hist_1','hist_2',
                    'hist_3','hist_4','hist_5','hist_6','hist_7','hist_8'], axis = 1)
new_df1.head(5)
```

Out[5]:

	Device id	Device name	time	dayOfWeek	max_total	left_avg_mins (dwell time)	left	year	month	location	daynumber
0	014d9e	Botanical Gardens - conservatory entrance	2021-08-25T06:30:00+05:30	Wednesday	0.0	0.000000	0.0	2021	Aug	-38.1494367,144.3774611	25
1	014db4	rippleside north end near gym	2021-08-25T06:30:00+05:30	Wednesday	63.0	7.750000	154.0	2021	Aug	-38.1274401,144.3549584	25
2	014dc6	Little Malop and James - CCTV/wifi cabinet	2021-08-25T06:30:00+05:30	Wednesday	125.0	16.501127	177.0	2021	Aug	-38.1479277,144.3588385	25
3	hawk-013a20	Busport Project office	2021-08-25T06:30:00+05:30	Wednesday	0.0	0.000000	0.0	2021	Aug	-38.1447631,144.3595969	25
4	hawk-013a21	155 Mercer Foyer	2021-08-25T06:30:00+05:30	Wednesday	1.0	0.000000	0.0	2021	Aug	-38.1458692,144.3581138	25

In [6]:
```
# renaming the columns of regularupdated_cleaned_wifi_data dataset as per cleaned_wifi_data dataset
new_df1.rename(columns = {'max_total':'total','left_avg_mins (dwell time)':'left_avg','dayOfWeek':'dayofweek',
                          'Device id':'dev_id','Device name':'name','time':'date','daynumber':'day'},inplace = True)
```

In [7]:
```
# removing the extra columns in cleaned_wifi_data dataset which are not present in regularupdated_cleaned_wifi_data dataset
new_df = df.drop(['updatetime','hour'], axis = 1)
new_df.head(5)
```

Out[7]:

	dev_id	name	date	total	left	left_avg	year	month	dayofweek	day	location
0	hawk-013a4f	City Hall	2019-01-03T10:03:20+05:30	27.0	14.0	1.0	2019.0	Jan	Thursday	3.0	-38.1474393,144.3576924
1	hawk-013a4f	City Hall	2019-01-03T10:03:20+05:30	33.0	13.0	1.0	2019.0	Jan	Thursday	3.0	-38.1474393,144.3576924
2	hawk-013a51	Myer South side	2019-01-03T10:03:20+05:30	72.0	23.0	1.0	2019.0	Jan	Thursday	3.0	-38.14762,144.3622749
3	hawk-013a23	LaPorcheta	2019-01-03T10:03:20+05:30	16.0	14.0	1.0	2019.0	Jan	Thursday	3.0	-38.1469395,144.3599357
4	hawk-013a23	LaPorcheta	2019-01-03T10:03:20+05:30	19.0	10.0	3.0	2019.0	Jan	Thursday	3.0	-38.1469395,144.3599357

In [8]:
```
# checking wheather the columns are matching for both dataframes

print("Updated attributes of cleaned_wifi_data dataset")
for col in new_df.columns:
    print(col)

print("\n Updated attributes of regularupdated_cleaned_wifi_data dataset")
for col in new_df1.columns:
    print(col)
```

```
Updated attributes of cleaned_wifi_data dataset
dev_id
name
date
total
left
left_avg
year
month
dayofweek
day
location

 Updated attributes of regularupdated_cleaned_wifi_data dataset
dev_id
name
date
dayofweek
total
left_avg
left
year
month
location
day
```

Merging operation of both cleaned datsets

In [10]:
```
append_df = new_df.append(new_df1, ignore_index = True)
append_df
```

Out[10]:

	dev_id	name	date	total	left	left_avg	year	month	dayofweek	day	location
0	hawk-013a4f	City Hall	2019-01-03T10:03:20+05:30	27.000000	14.000000	1.000000	2019.0	Jan	Thursday	3.0	-38.1474393,144.3576924
1	hawk-013a4f	City Hall	2019-01-03T10:03:20+05:30	33.000000	13.000000	1.000000	2019.0	Jan	Thursday	3.0	-38.1474393,144.3576924
2	hawk-013a51	Myer South side	2019-01-03T10:03:20+05:30	72.000000	23.000000	1.000000	2019.0	Jan	Thursday	3.0	-38.14762,144.3622749
3	hawk-013a23	LaPorcheta	2019-01-03T10:03:20+05:30	16.000000	14.000000	1.000000	2019.0	Jan	Thursday	3.0	-38.1469395,144.3599357
4	hawk-013a23	LaPorcheta	2019-01-03T10:03:20+05:30	19.000000	10.000000	3.000000	2019.0	Jan	Thursday	3.0	-38.1469395,144.3599357
...	...	...	...	...	...	...	...	...	...	...	...
1059214	014de9	High Street , Belmont wifi/cctv cabinet	2021-12-07T11:30:00+05:30	202.000000	494.000000	13.872920	2021.0	Dec	Tuesday	7.0	-38.1746141,144.3430678
1059215	hawk-013a33	Ryrie/Moorabool Outside	2021-12-07T11:30:00+05:30	33.000000	45.000000	9.166667	2021.0	Dec	Tuesday	7.0	-38.1489155,144.3584489
1059216	meraki-e0-cb-bc-35-ae-29	Lt Malop West	2021-12-07T11:30:00+05:30	31.190542	67.596568	6.163429	2021.0	Dec	Tuesday	7.0	-38.147929,144.358854
1059217	meraki-e0-cb-bc-35-bb-14	Waterfront - Pier	2021-12-07T11:30:00+05:30	31.190542	67.596568	6.163429	2021.0	Dec	Tuesday	7.0	-38.143364,144.361926
1059218	meraki-e0-cb-bc-35-be-84	Lt Malop Central	2021-12-07T11:30:00+05:30	31.190542	67.596568	6.163429	2021.0	Dec	Tuesday	7.0	-38.148573,144.361279

1059219 rows × 11 columns

Data processing

Google Maps API Service Integration

Now, we have integrated 'Google Maps API' service and used 'Places API' service on cleaned merged dataset to fetch location of Wi-Fi routers & building types (Hospitals, Libraries, Malls, Gyms & Restaurants) in a 1 km radius to monitor total pedestrian count & number of citizens left during the period. Furthermore, we have also mapped those locations on google maps using their respective coordinates attached to each router to verify the authenticity of the data.

```
In [32]: df
```

Out[32]:

location	latitude	longitude	hospitals_count	libraries_count	gyms_count	restaurants_count	malls_count	url
4.3576924	-38.1474393	144.3576924	20	9	14	20	20	https://www.google.com/maps/place/-38.1474393,...
4.3576924	-38.1474393	144.3576924	20	9	14	20	20	https://www.google.com/maps/place/-38.1474393,...
4.3622749	-38.14762	144.3622749	19	7	12	20	20	https://www.google.com/maps/place/-38.14762,14...
4.3599357	-38.1469395	144.3599357	19	10	15	20	20	https://www.google.com/maps/place/-38.1469395,...
4.3599357	-38.1469395	144.3599357	19	10	15	20	20	https://www.google.com/maps/place/-38.1469395,...
...	...	...	...	...	...	...	...	...
4.3430678	-38.1746141	144.3430678	0	3	8	20	2	https://www.google.com/maps/place/-38.1746141,...
4.3584489	-38.1489155	144.3584489	19	10	19	20	20	https://www.google.com/maps/place/-38.1489155,...
44.358854	-38.147929	144.358854	19	10	14	20	20	https://www.google.com/maps/place/-38.147929,1...
44.361926	-38.143364	144.361926	20	8	13	20	20	https://www.google.com/maps/place/-38.143364,1...
44.361279	-38.148573	144.361279	19	9	14	20	20	https://www.google.com/maps/place/-38.148573,1...

Reverse Geocoding & creating Lockdown stages

After integrating 'Google Maps API' service in order to fetch location of Wi-Fi routers & building types (Hospitals, Libraries, Malls, Gyms & Restaurants), we have used 'Reverse Geocoding (using geopy library)' to obtain address, post-codes & suburbs. We also created 8 lockdown stages & mapped the lockdown stages of Victoria using the lockdown stages information to each 'device ID' or router to analyze the impact of pedestrian count in different suburbs and buildings using Wi-Fi routers in the lockdown stages by merging the 'reverse geocoded' data & 'lockdown stage' data to the previously implemented dataset consisting of integration of 'Google Maps API' services.

Using Reverse Geocoding to extract 'addresses', 'postcodes' & 'suburbs'

In [4]:
```
# extracting the addresses using reverse geocoding
from geopy.geocoders import Nominatim
geolocator = Nominatim(user_agent="test_app")
full_address=[]
for i in range(len(coords)):
    location = geolocator.reverse(coords[i])
    address=location.raw['display_name']
    full_address.append(address)
```

In [5]:
```
coord_df['address']= full_address
coord_df
```

Out[5]:

	coordinates	latitude	longitude	address
0	-38.1474393,144.3576924	-38.147439	144.357692	Gheringhap Street, Geelong CBD, City of Greate...
1	-38.14762,144.3622749	-38.147620	144.362275	Myer, Malop Street, Geelong, City of Greater G...
2	-38.1469395,144.3599357	-38.146940	144.359936	WDEA Works, 37, Malop Street, Geelong, City of...
3	-38.1489155,144.3584489	-38.148916	144.358449	Geelong CBD, City of Greater Geelong, Victoria...
4	-38.1458692,144.3581138	-38.145869	144.358114	155, Mercer Street, Geelong, City of Greater G...
5	-38.1691458,144.3511856	-38.169146	144.351186	Barwon Valley Activity Centre, 1, Barwon Heads...
6	-38.1220226,144.3591414	-38.122023	144.359141	Swinburne Street, Rippleside, City of Greater ...

In [6]:
```
#Breakdown of address
address = location.raw['address']
print(address)
```

```
{'road': 'The Esplanade', 'village': 'Indented Head', 'municipality': 'City of Greater Geelong', 'state': 'Victoria', 'postcod
e': '3223', 'country': 'Australia', 'country_code': 'au'}
```

In [9]:
```
# using regex to filter out the postcode and suburbs from the addresses
coord_df['postcode'] = coord_df['address'].str.extract('([0-9][0-9][0-9][0-9])')
coord_df['suburb'] = coord_df['address'].str.extract('.*, (.*), (.*), Victoria')
coord_df
```

Out[9]:

	coordinates	latitude	longitude	address	postcode	suburb
0	-38.1474393,144.3576924	-38.147439	144.357692	Gheringhap Street, Geelong CBD, City of Greate...	3218	Geelong CBD
1	-38.14762,144.3622749	-38.147620	144.362275	Myer, Malop Street, Geelong, City of Greater G...	3218	Geelong
2	-38.1469395,144.3599357	-38.146940	144.359936	WDEA Works, 37, Malop Street, Geelong, City of...	3218	Geelong
3	-38.1489155,144.3584489	-38.148916	144.358449	Geelong CBD, City of Greater Geelong, Victoria...	3218	NaN
4	-38.1458692,144.3581138	-38.145869	144.358114	155, Mercer Street, Geelong, City of Greater G...	3218	Geelong
5	-38.1691458,144.3511856	-38.169146	144.351186	Barwon Valley Activity Centre, 1, Barwon Heads...	3216	Belmont
6	-38.1220226,144.3591414	-38.122023	144.359141	Swinburne Street, Rippleside, City of Greater ...	3215	Rippleside
7	-38.1496014,144.3608612	-38.149601	144.360861	Bubble Bar, 188, Ryrie Street, Geelong, City o...	3218	Geelong
8	-38.1274401,144.3549584	-38.127440	144.354958	Rippleside Park Toilet Block, Rippleside Park ...	3215	Rippleside

In [21]:
```
df
```

Out[21]:

hospitals_count	libraries_count	gyms_count	restaurants_count	malls_count	url	address	postcode	suburb
20	9	14	20	20	https://www.google.com/maps/place/-38.1474393,...	Gheringhap Street, Geelong CBD, City of Greate...	3218	Geelong CBD
20	9	14	20	20	https://www.google.com/maps/place/-38.1474393,...	Gheringhap Street, Geelong CBD, City of Greate...	3218	Geelong CBD
19	7	12	20	20	https://www.google.com/maps/place/-38.14762,14...	Myer, Malop Street, Geelong, City of Greater G...	3218	Geelong
19	10	15	20	20	https://www.google.com/maps/place/-38.1469395,...	WDEA Works, 37, Malop Street, Geelong, City of...	3218	Geelong
19	10	15	20	20	https://www.google.com/maps/place/-38.1469395,...	WDEA Works, 37, Malop Street, Geelong, City of...	3218	Geelong
...	...	...	...	...	...	...	...	...
0	3	8	20	2	https://www.google.com/maps/place/-38.1746141,...	Busy Bee Fabrics, Belmont Centreway, Belmont, ...	3216	Belmont
19	10	19	20	20	https://www.google.com/maps/place/-38.1489155,...	Geelong CBD, City of Greater Geelong,	3218	Geelong CBD

In 'Lockdown_stage', there are 8 stages:

1st - Lockdown-1 (Tuesday 31st March 2020 to Tuesday 12th May 2020),
2nd - Lockdown-2 (Thursday 9th July 2020 to Tuesday 27th October 2020)
3rd - Lockdown-3 (Saturday 13th February 2021 to Wednesday 17th February 2021)
4th - Lockdown-4 (Friday 28th May 2021 to Thursday 10th June 2021)
5th - Lockdown-5 (Friday 16th July 2021 to Tuesday 27th July 2021)
6th - Lockdown-6 (5 August 2021 – 21 October 2021)
7th - open (22 October 2021- 19th November 2021)
8th - others (dates where there is no lockdown except 7th)

In [27]:
```
#set lockdown duration (lockdown-1 to 6 & including Post lockdown stage & others)
date_start1, date_end1 = datetime(2020, 3, 31), datetime(2020, 5, 12)  # Lockdown Stage-1
date_start2, date_end2 = datetime(2020, 7, 9), datetime(2020, 10, 27)  # Lockdown Stage-2
date_start3, date_end3 = datetime(2021, 2, 13), datetime(2021, 2, 17)  # Lockdown Stage-3
date_start4, date_end4 = datetime(2021, 5, 28), datetime(2021, 6, 10)  # Lockdown Stage-4
date_start5, date_end5 = datetime(2021, 7, 16), datetime(2021, 7, 27)  # Lockdown Stage-5
date_start6, date_end6 = datetime(2021, 8, 5), datetime(2021, 10, 21)  # Lockdown Stage-6
date_start7, date_end7 = datetime(2021, 10, 22), datetime(2021, 12, 7) # Post Lockdown Stage
lockdown = []
for lock in date_df['date']:
    # checking for date in range
    if lock >= date_start1 and lock <= date_end1:
        output = 'Lockdown-1'
    elif lock >= date_start2 and lock <= date_end2:
        output = 'Lockdown-2'
    elif lock >= date_start3 and lock <= date_end3:
        output = 'Lockdown-3'
    elif lock >= date_start4 and lock <= date_end4:
        output = 'Lockdown-4'
    elif lock >= date_start5 and lock <= date_end5:
        output = 'Lockdown-5'
    elif lock >= date_start6 and lock <= date_end6:
        output = 'Lockdown-6'
    elif lock >= date_start7 and lock <= date_end7:
        output = 'Open'
    else:
        output = 'others'
    lockdown.append(output)
```

In [28]:
```
df['Lockdown_stage'] = lockdown
```

In [29]:
```
df
```

Out[29]:

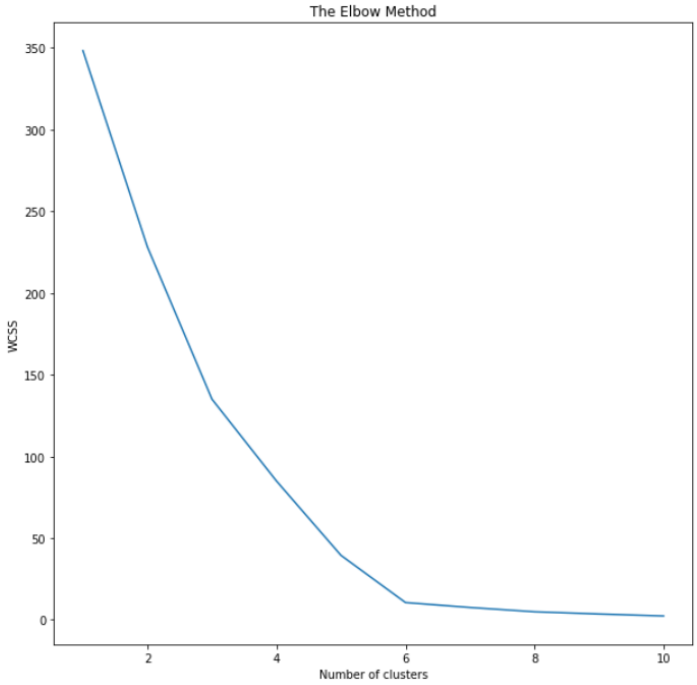
libraries_count	gyms_count	restaurants_count	malls_count	url	address	postcode	suburb	Lockdown_stage
9	14	20	20	https://www.google.com/maps/place/-38.1474393,...	Gheringhap Street, Geelong CBD, City of Greate...	3218	Geelong CBD	others
9	14	20	20	https://www.google.com/maps/place/-38.1474393,...	Gheringhap Street, Geelong CBD, City of Greate...	3218	Geelong CBD	others
7	12	20	20	https://www.google.com/maps/place/-38.14762,14...	Myer, Malop Street, Geelong, City of Greater G...	3218	Geelong	others
10	15	20	20	https://www.google.com/maps/place/-38.1469395,...	WDEA Works, 37, Malop Street, Geelong, City of...	3218	Geelong	others
10	15	20	20	https://www.google.com/maps/place/-38.1469395,...	WDEA Works, 37, Malop Street, Geelong, City of...	3218	Geelong	others
...	...	...	...	...	...	...	...	...
3	8	20	2	https://www.google.com/maps/place/-38.1746141,...	Busy Bee Fabrics, Belmont Centreway, Belmont, ...	3216	Belmont	Open
10	19	20	20	https://www.google.com/maps/place/-38.1489155,...	Geelong CBD, City of Greater Geelong	3218	Geelong CBD	Open

Later, saved the dataframe to a CSV file for visualization in Tableau.

Clustering & Predictive Modelling

Clustering

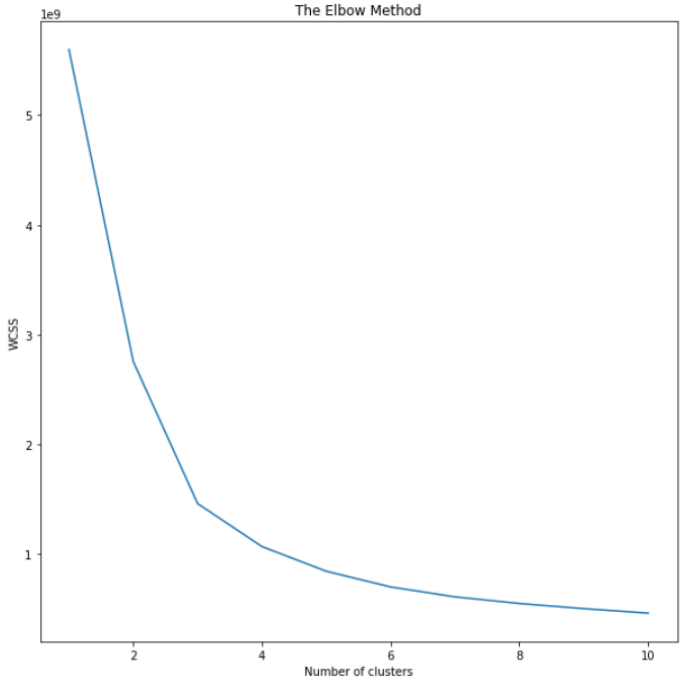
After that we dropped some irrelevant columns & did ‘label encoding’ onto the required features for clustering part. We researched & performed ‘Geo-Spatial’ Clustering on the dataset using ‘k-means’ clustering in order to learn the characteristics of our dataset visually. Most coordinate points were superimposing on each other as the distance between them is very less. Done ‘PCA’ for dimensionality reduction & to avoid overfitting but it wasn’t needed as we were only considering ‘latitude & longitude’ column for geo-spatial clustering & for normal clustering, we were having information loss as some important features were discarded & for time series prediction it may give low accuracy, so we dropped PCA. We found the optimal value of ‘k’ in k-means clustering using the ‘Elbow method’ and also used a ‘kneed’ library to programmatically find the optimal value of k, which was found to be ‘5’ in case of Geo-Spatial Clustering.




```
In [37]: kl = KneeLocator(range(1, 11), wcss, curve="convex", direction="decreasing")
         kl.elbow
```

```
Out[37]: 5
```

After that, we have done Clustering using all the features on the dataset & again found the optimal value of 'k' using 'Elbow Method' & 'kneed' library which was found to be '3', which took a very long time as we were dealing with more than 1 million records. Later, we analyzed the clusters & clustered our dataset into 'Dense', 'Sparse' & 'Normal' groups on the basis of Wi-Fi usage. Later, saved the dataframe to a CSV file for visualization & input to predictive modelling.




```
In [44]: kl = KneeLocator(range(1, 11), wcss, curve="convex", direction="decreasing")
         print('Best K-value (using KneeLocator) utilizing k-means:',kl.elbow)
```

```
Best K-value (using KneeLocator) utilizing k-means: 3
```

Boosting & other Supervised Machine Learning Models & Neural Network (LSTM)

After Clustering, we have handled data imbalance for multiclass classification using 'SMOTE (Synthetic Minority Oversampling Technique)' & 'RandomUnderSampler (for down-sampling)' on the clustered dataset. The input was later fed to Boosting (XGBoost) & other supervised Machine learning models (Decision Tree, KNN & SVM) which gave very great results overall. Random Forest was identified as the best model for Wi-Fi prediction for 6th & 7th Dec 2021 Wi-Fi Dataset at an overall accuracy of 97% and F1-Score of 98%. These results also have been shared with the Data visualization team for Visualization in Tableau. For the time series prediction, we got around 48% accuracy using LSTM.

	precision	recall	f1-score	support
0.0	1.00	0.97	0.98	14101
1.0	0.93	1.00	0.96	13955
2.0	1.00	0.95	0.98	13902
accuracy			0.97	41958
macro avg	0.98	0.97	0.97	41958
weighted avg	0.98	0.97	0.97	41958

Visualization for Pedestrian Data (using Tableau)

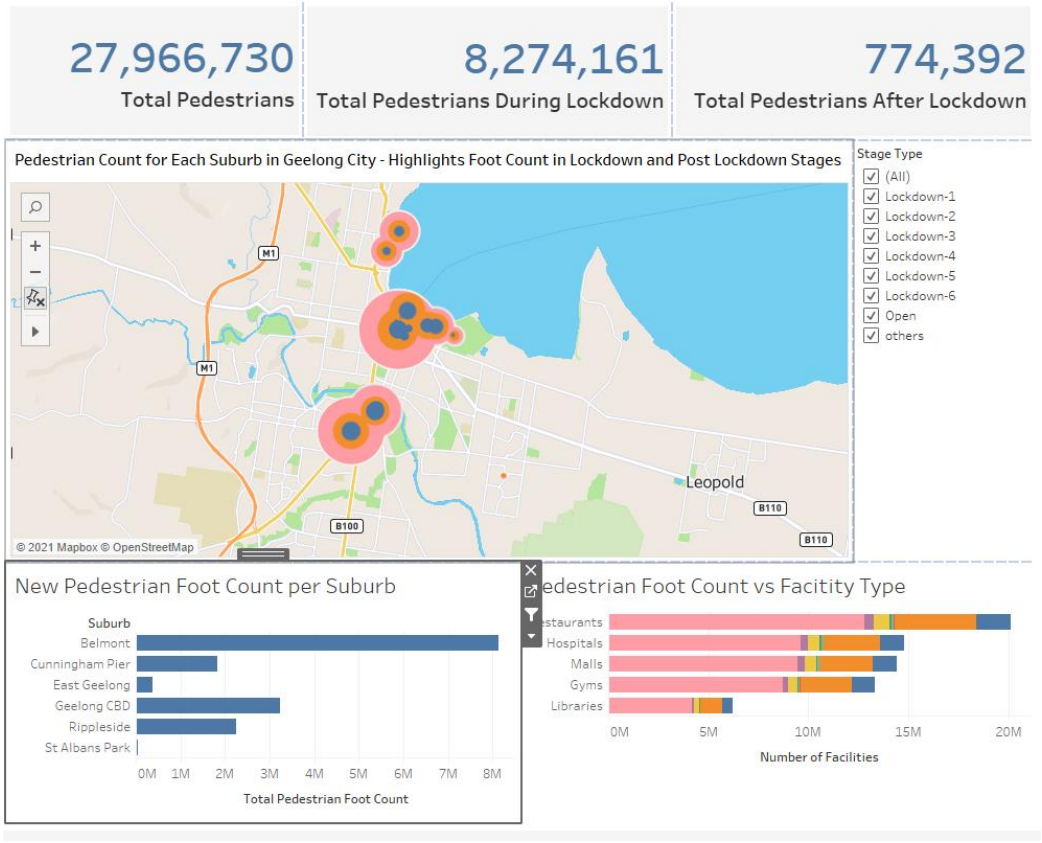
For the Pedestrian problem, it shows the pedestrian count for Geelong in normal day is 27,966,730 and the pedestrian count for Geelong in lockdown is 8,274,161. In addition, the total pedestrian count for Geelong after lockdown is 774,392. However, the smallest pedestrian count is in Belmont (8,123,258 in total). For this maps, it clearly shows the distribution of pedestrian for different regions in different lockdown stages. It shows the circle bigger is, the greater number of pedestrians.

Suggestions for the next squad:

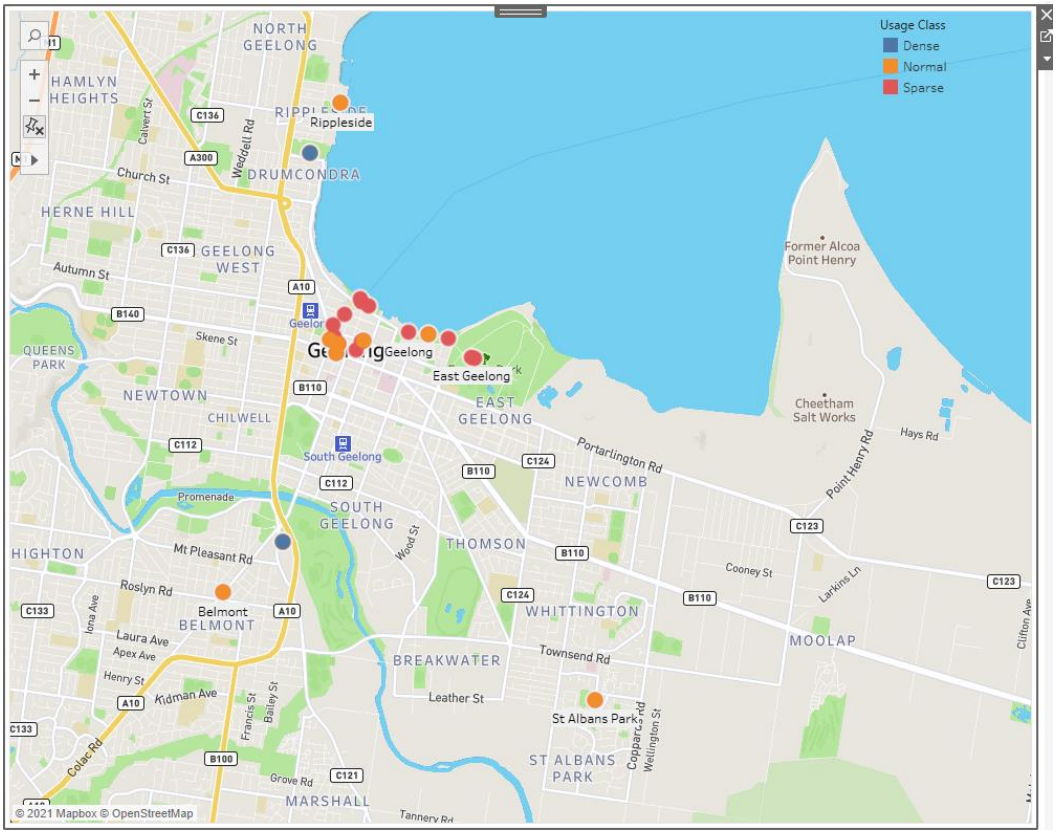
In the iteration, we will continue to optimize the data prediction by the several following ways:
- Making data live or pipelining the live data to website.
- Making the data richer with new records.
- Establishing some conclusive correlations for features for implementing prediction models.

NOTE

Prepared by:	*Anyu Xu*
Point of Contact:	*Anyu Xu*, *xuany@deakin.edu.au*
Team Members:	*Anyu Xu*, *Chris Chen* *Sahit Mantripragada*



Visualization for Wi-Fi Data using Tableau



For this map, it clearly shows the distribution of Wi-Fi using situation for different suburb in Geelong. The orange colour shows the suburb has Normal dense situation, and the blue colour shows the suburb has dense situation, and the red colour shows the suburb has sparse situation.

Suggestions for the next squad:

In the iteration, we will continue to optimize the data prediction by the several following ways:
- Making data live or pipelining the live data to website.
- Making the data richer with new records.

Web Development Team

Web development team worked on the frontend with the goal of making the website more industry-standard and modern. Technologies including **HTML, Bootstrap, JavaScript, CSS, SaSS, DJango** are used for web development team deliveries.

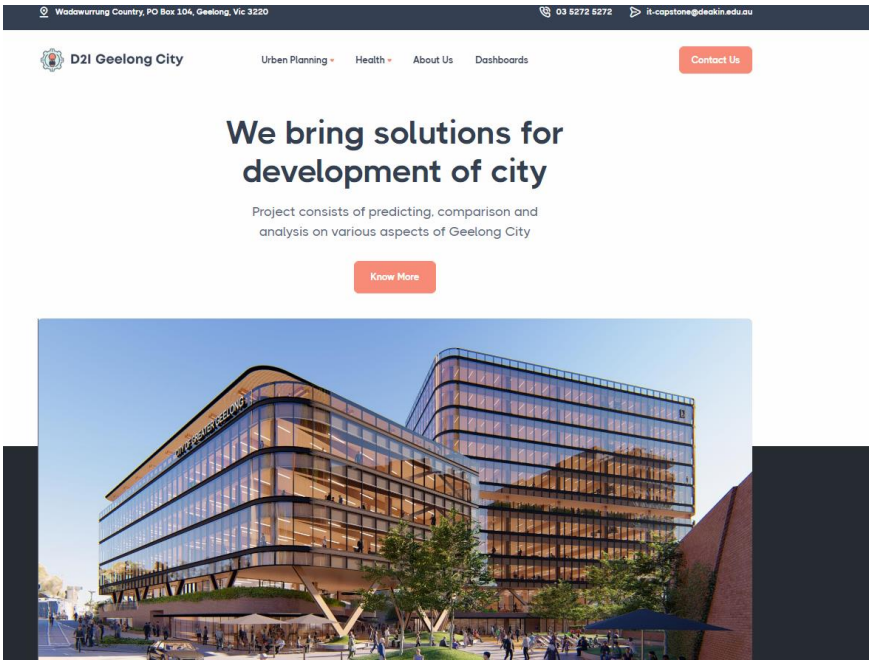
NOTE

Prepared by:	*Abdullah Abdulaziz M Algamdi*
Point of Contact:	*Abdullah Abdulaziz M Algamdi*, *aalgamdi@deakin.edu.au*
Team Members:	*Rakshith Rachenahalli Bettegowda*, *Abdullah Abdulaziz M Algamdi*, *Saleh Mohammed M Alharbi*

The main deliverables for the web development team are-

Enhance Web Pages

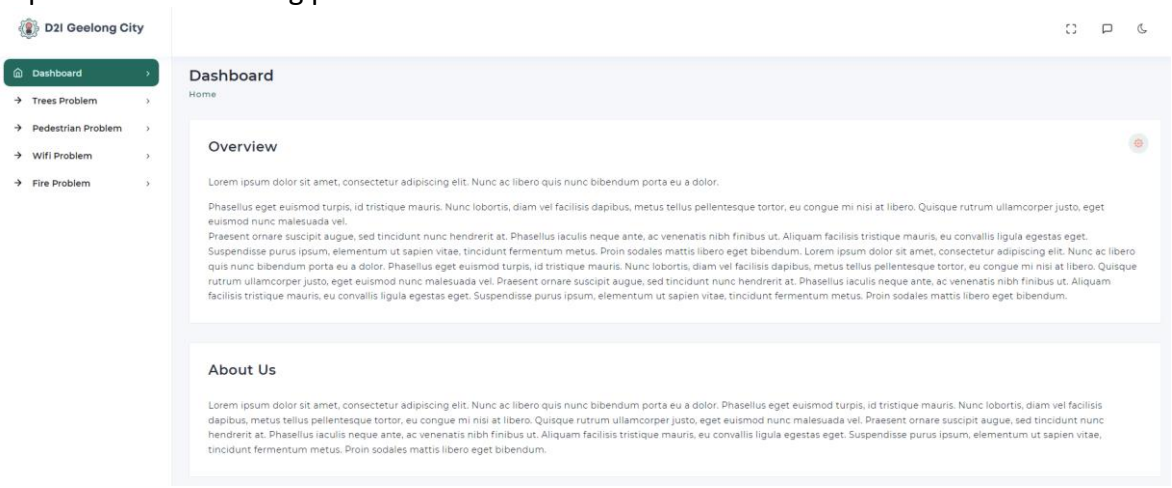
- Web pages and dashboards pertaining to urban planning and health services have been enhanced.



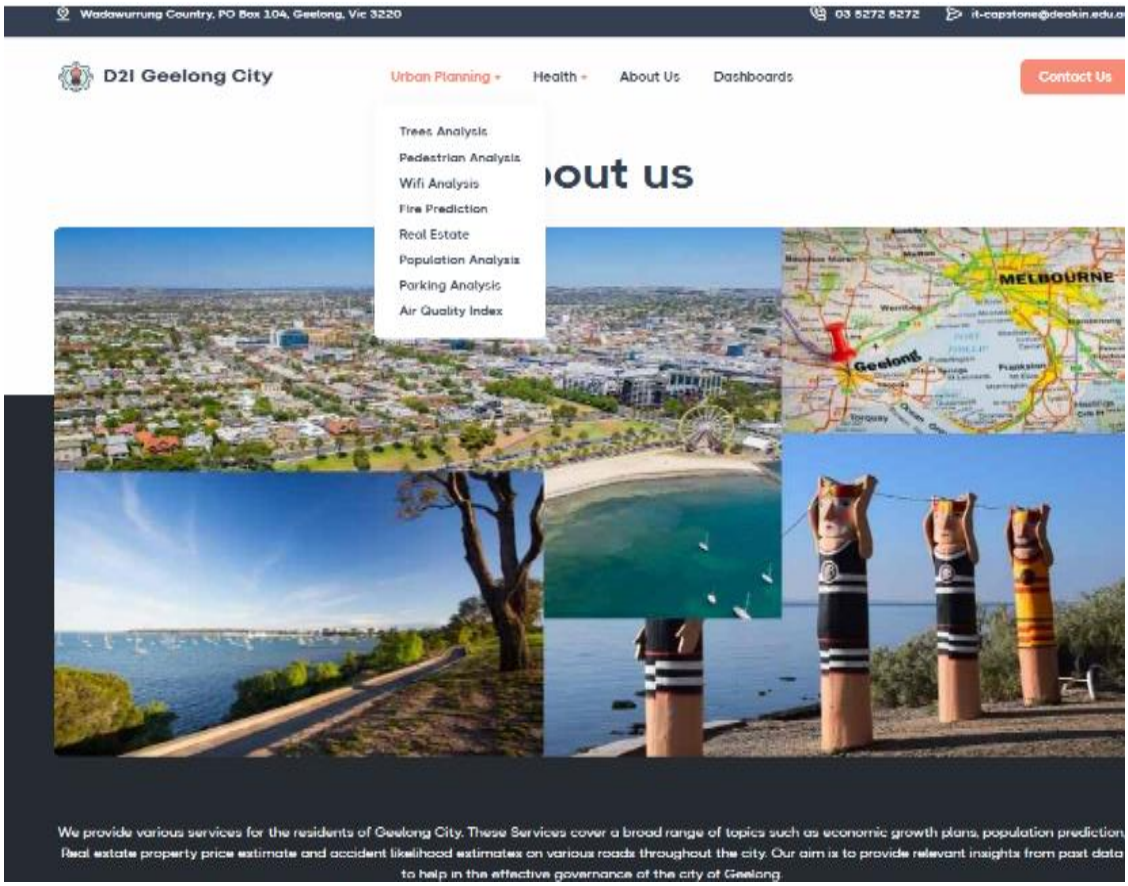
- All the broken links have been identified and fixed.
- Unnecessary JavaScript and inline CSS are removed.
- Web pages are optimized for all devices.

Dashboards

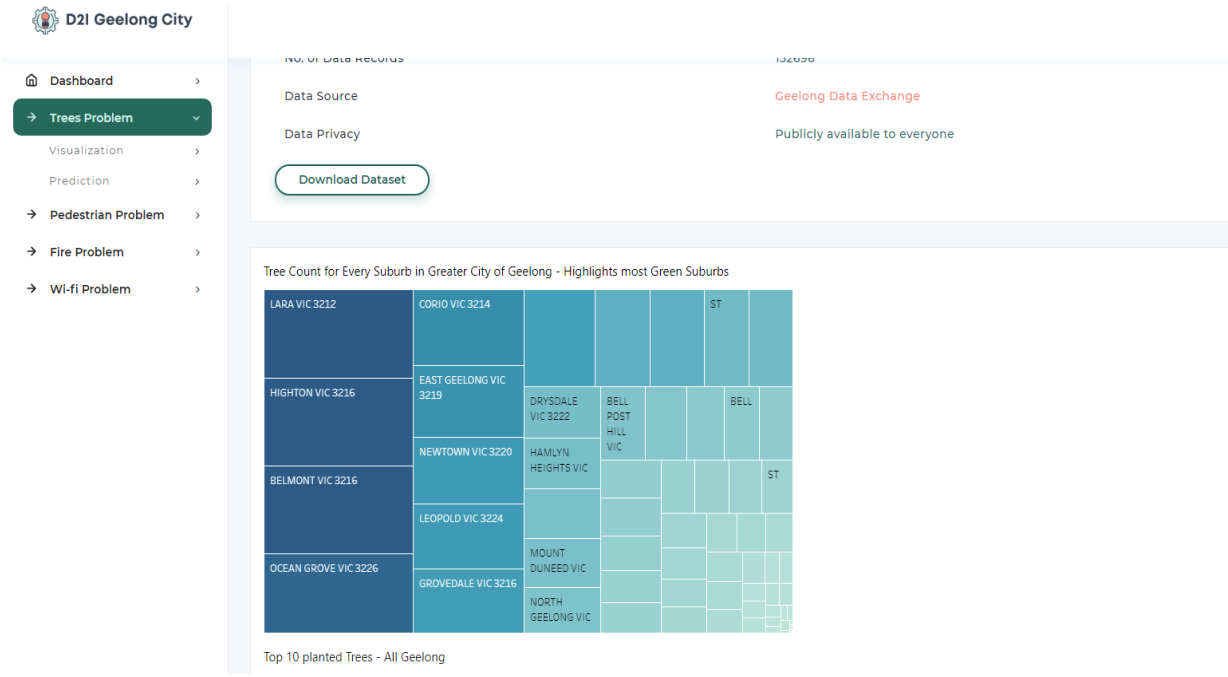
- New dashboards have been designed for four problems that includes Wi-Fi, Pedestrian, Tree and Fire problems. Remaining problems are linked to old dashboards.



- Dashboards are designed to improve the user interface. The dashboards are built with the use of **advanced CSS, bootstraps, and additional resources**.

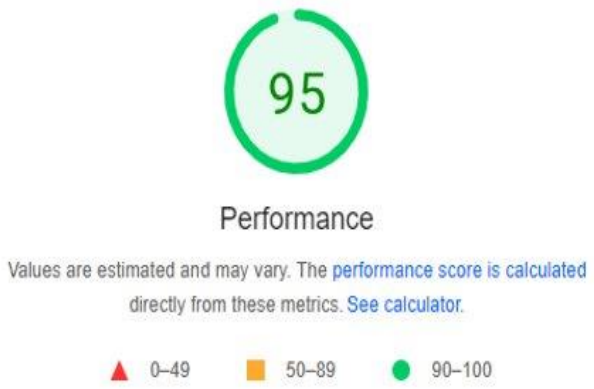


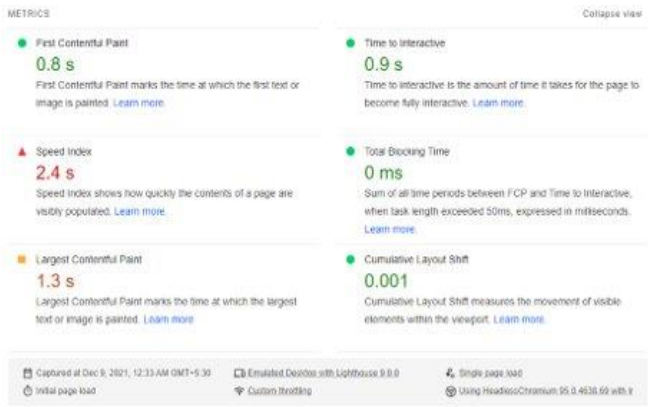
- Dashboards displays all the issues that are being addressed and improved during this trimester.



Performance

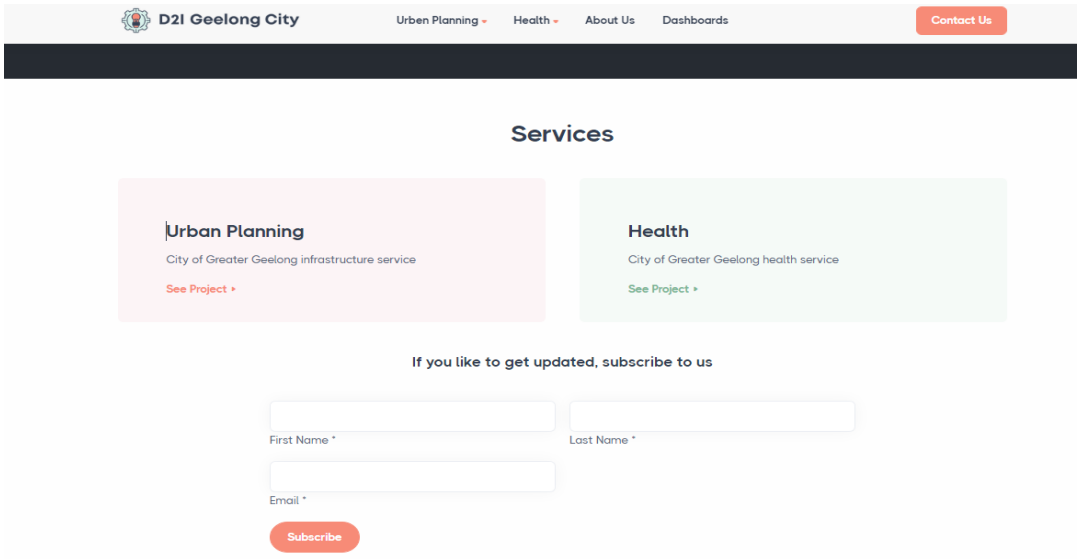
- Have text at the top of the page for faster loading speed. The website's overall performance is evaluated in a google-managed page-performance website and found that the results are good.





New Mailing subscription with 2F Authentication

- A mailing subscription system with 2 factor authentication has been implemented where subscription is confirmed only when user verify the link sent to his email id in a Welcome email. User subscription is managed by ‘MailChimp’ automation.



Django based portal hosted on localhost

- A new Django based portal has been developed which is currently hosted on localhost.
- To start the portal run following command at console.
  python manage.py runserver

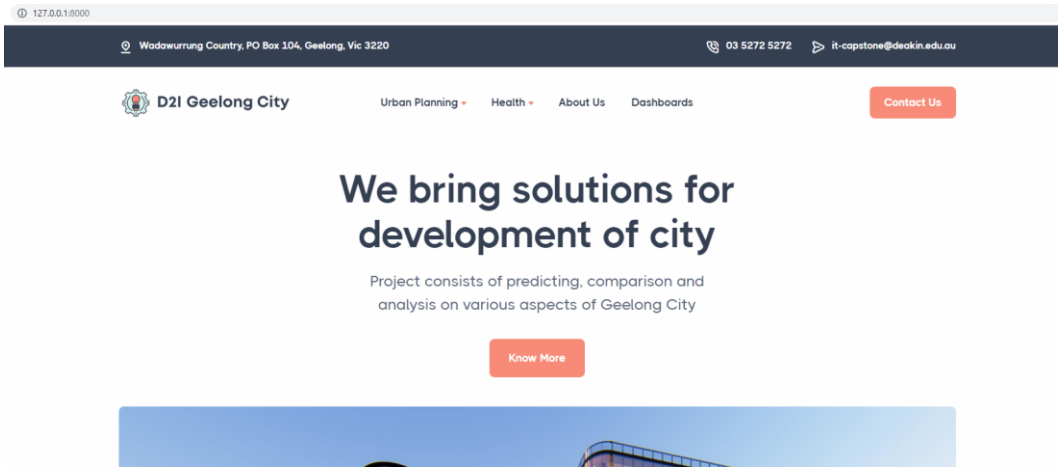
  Refer User Manual section for setting the environment.

```
c:\D2I-Geelong>env\Scripts\activate

(env) c:\D2I-Geelong>python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

C:\D2I-Geelong\env\lib\site-packages\sklearn\base.py:329: UserWarning: Trying to unpickle estimator StandardScaler from
version 0.22.2.post1 when using version 0.23.0. This might lead to breaking code or invalid results. Use at your own ris
k.
  warnings.warn(
System check identified no issues (0 silenced).
December 16, 2021 - 16:36:24
Django version 3.1.1, using settings 'D2i-Geelong-T3-2021-WebDevelopment.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

- After that open browser and type http://localhost:8000 or http://127.0.0.:8000 in url

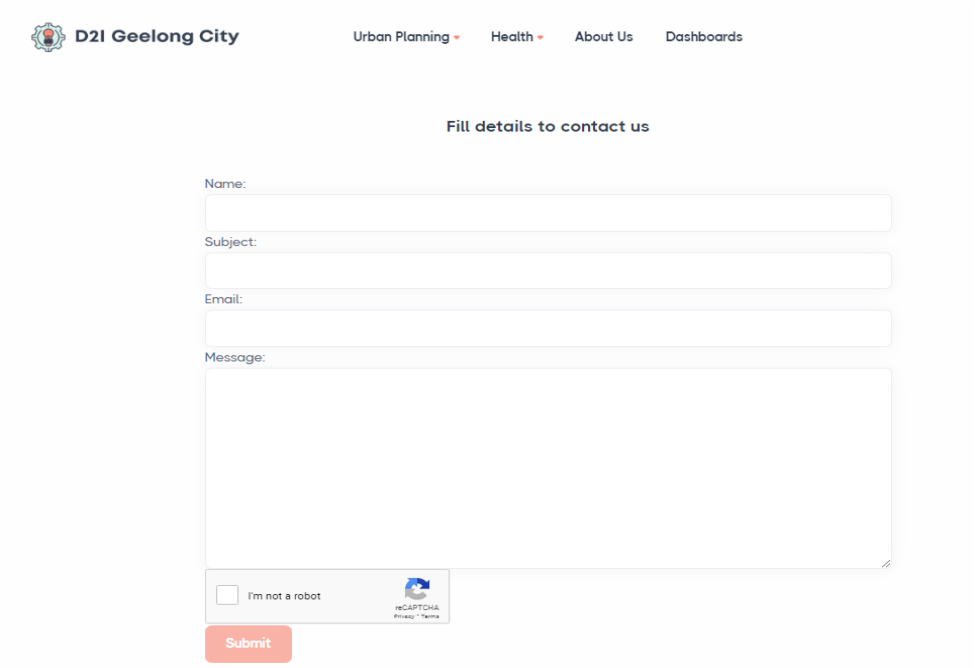


“Contact Us” Form

- A more secure “Contact Us” form has been implemented by using the Django forms. The form is protected with google re-captcha version 2.
- Submit gets enabled only when captcha is verified. When user clicks on submit button, a mail with all the provided information is sent to a back-office email id. Credentials of the back-office email id are:

  Email: D2I.Geelong@gmail.com
  Password: D2IT3Geelong2021

  Google re-captcha used in contact us form are registered against the same email id.

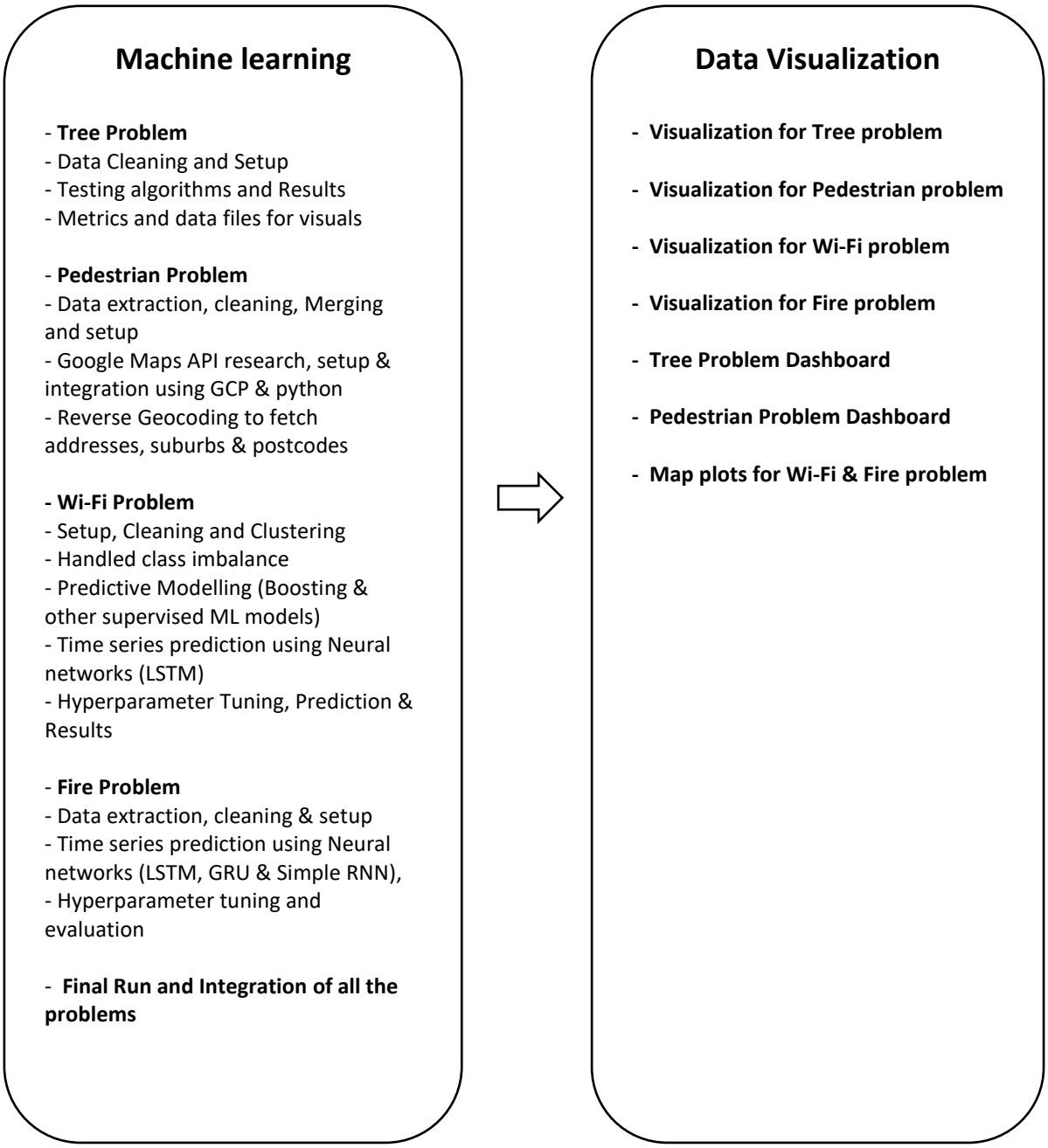


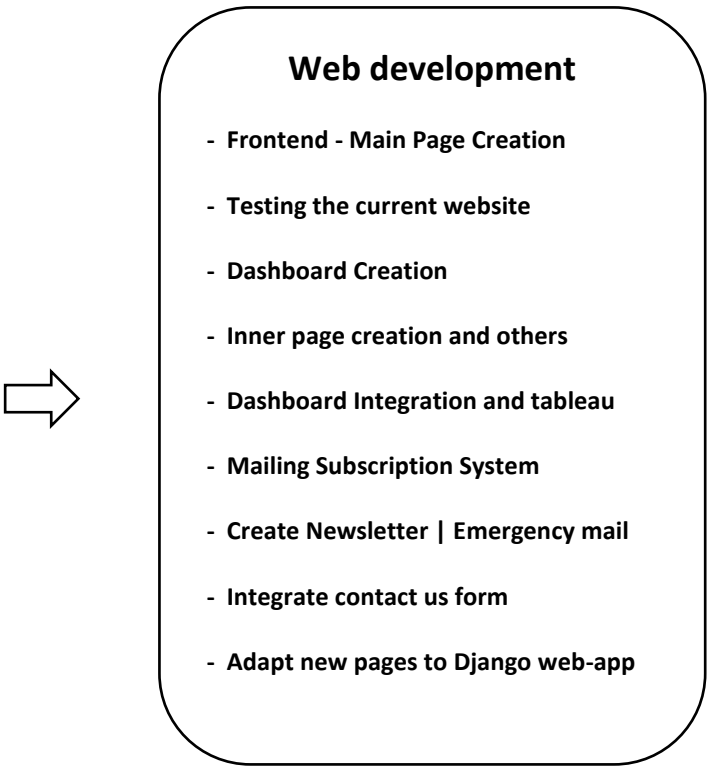
Deployment on Netlify Platform

- Webpages source code is pushed to GitHub. The credential used for github repository are
  Repository URL: https://github.com/D2i-GeelongCity?tab=repositories
  Email: d2i.geelongcity@gmail.com
  Username: D2i-GeelongCity
  Password: haha@098@haha
- Github repository is linked to Netlify account.
- URL of deployed site is https://d2i-geelong-city.netlify.app

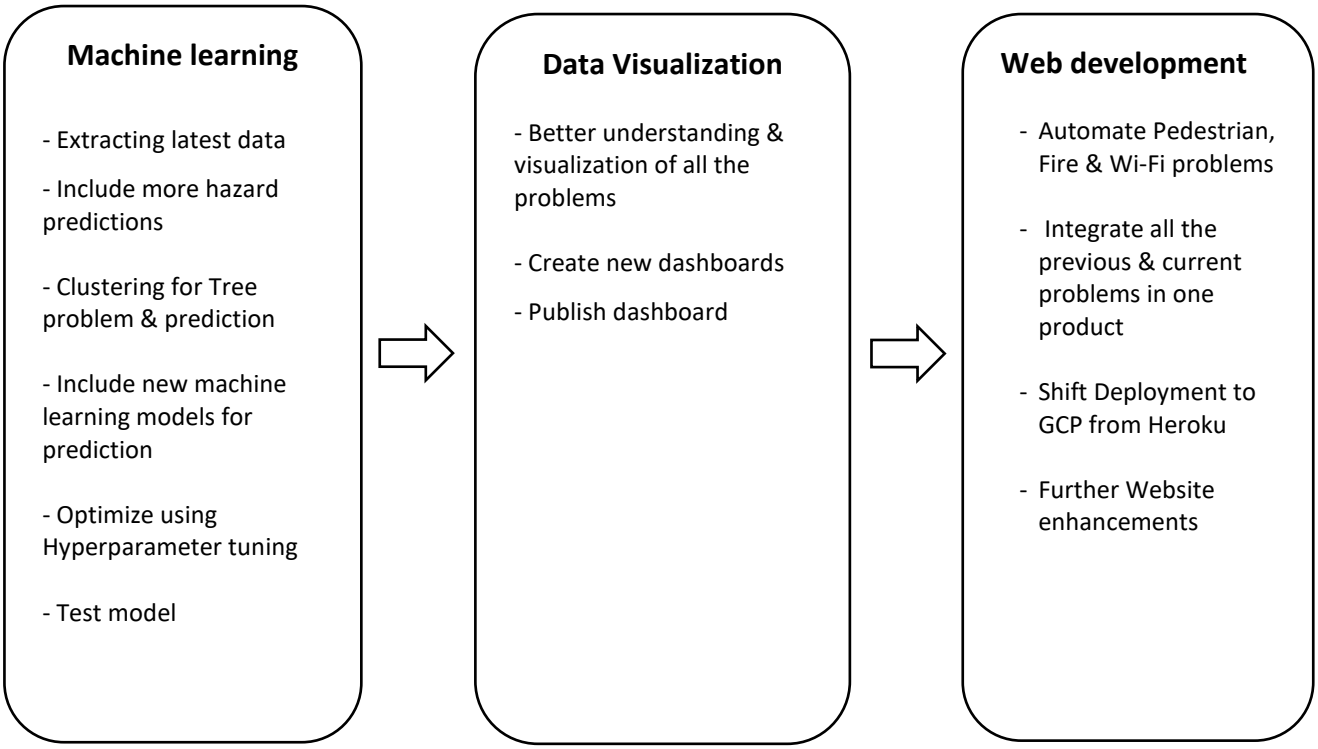
5. Roadmap

The following roadmap relates to trimester 3, 2021 & is available on the project roadmap board on Trello. That roadmap is summarised as below:





For the further development in future iterations, we lay out a suggested 'roadmap', details are also showcased below:



6. Open Issues

During the T3 Trimester, our team faced several challenges on Machine Learning, Data Visualization & Web Development front which required a great deal of research & debugging in the process, we achieved & showcased exceptional results in the end. Along the way we had some issues related to technology which we acknowledged & planned to mitigate in the next semester, these issues are illustrated below:

Tree Problem

- We can further increase the overall accuracy of our model by venturing into MLP (Multi-layer Perceptron) classifier.
- As there are huge number of classes at most 100, we can do clustering and group the entire problem in particular classes according to their respective features.
- Furthermore, we can do prediction on the clustered dataset to get better & efficient results.

Fire Problem

- We can add more features to the existing Fire problem to increase the generalizability of the model.
- We can also try certain tools like 'AutoML' for prediction exercise.

Overall, we need to work on '**automating**' the dataset of 'Wi-Fi/Pedestrian, Tree & Fire' problem which can then be used for prediction purpose efficiently. Also, we will try to **include new projects** related to different hazards in the upcoming trimester. We also need to work on **shifting the deployment** from **'Heroku' to 'GCP (Google Cloud Computing)' or 'AWS (Amazon Web Service)'** as per requirements. Lastly, we can also **introduce Bi-Directional LSTM** for better results in Trees & Fire problems. These project deliverables will depend on the client dependency and teamwork in the upcoming trimester.

7. Lessons Learned

Teamwork- The successful completion of the project is the perfect embodiment of our teamwork. For higher work efficiency, we assigned project tasks to junior members and senior members every week to ensure the smooth completion of the project. In addition, we discussed project deliverables every day, find out the problem, and help everyone complete the task in given time frame. In addition, our group held team meetings, Studio sessions & internal scrum meetings every week to discuss updates, ideas, results & work to be done in future iterations to achieve higher work efficiency and goals.

Time management- In this project, we all learned to use Trello cards efficiently to organize and plan the project. All members specified a particular deadline for their tasks and each sub-group leader reminded each member to make Trello cards beforehand. Therefore, all members effectively managed time to deliver iterative artifacts within the deadline.

Upskilling and solve problems- In week 2, we got together to discuss specific roles & tasks for our project. Each member holds a position based on their main IT background. The junior members learned various skills according to the recommendations of the senior members. After that, we learned different skills related to project deliverables from week 2 to week 6 in domain such as data extraction, data pre-processing, data visualization, etc., and then we used these skills to solve project deliverables expeditiously. Therefore, at the end we completed all the tasks of the project with high-quality products.

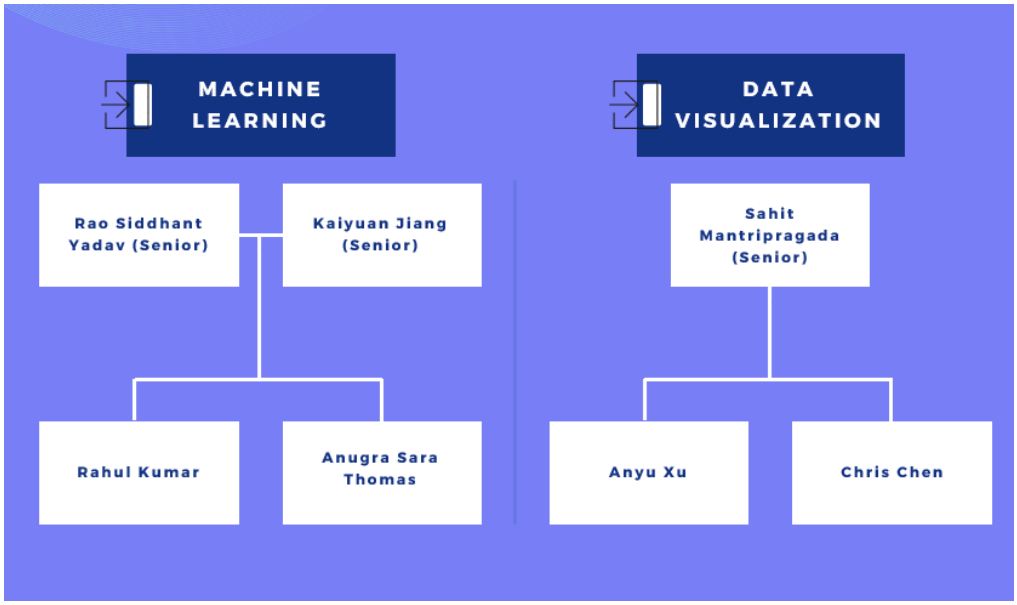
Academic mentors' feedback- Meeting with academic tutors weekly while getting recommendations & feedback-during the Studio sessions boosted our pace & accuracy of the project. We also updated our weekly work to our mentors and received feedbacks to help us overcome challenges to improve our work results. We discussed challenges and plans for next iteration every week. After that, we received mentors on how This helped us understand the challenges we faced & groomed us for our future endeavours.

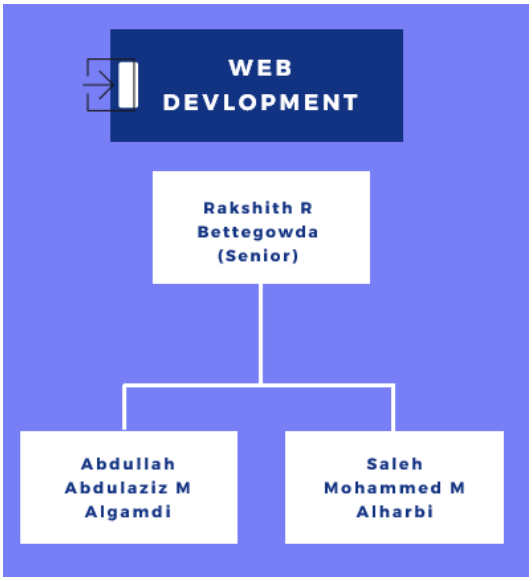
Project management- In this project, all members were involved in the completion of the project. According to the status of the project, member skills and time were allocated, prioritized, delegated and performed certain activities. These have effectively improved the skills of project management.

8. Product Development Life Cycle

8.1. New Tasks

In the light of Trimester 3 2021, the seniors in our squad came up with a total of 6 projects out of which 4 was selected by our client (**Trees, Pedestrian, Wi-Fi & Fire problems**) specifically for 'Geelong City' to make it internationally recognised as a 'clever and creative city-region that is forward looking, enterprising and adaptive, and cares for its people. The client also didn't have any dependency from his side. The squad was divided into 3-teams (Machine Learning, Data Visualization & Web Development team). Our squad consist a total of 10 members including seniors & juniors. The Machine Learning Team had 2 seniors and 2 juniors where each senior was allocated to a junior in order to work more efficiently. Both Data Visualization & Web-Development team had one senior & two juniors to prioritize workflow in a timely manner.





The seniors in our squad were very enthusiastic and we discussed daily updates of our project deliverables. The squad meets weekly in Squad meetings, internal scrum meetings & Studio Sessions to discuss updates, ideas, results & work to be done in future iterations which helped us achieve our target deliverables swiftly before the due date & made our project more efficient by following project scoping. In Client meetings, our squad acknowledged the client expectations and worked diligently throughout the trimester.

The seniors created pivotal **Trello cards** in sequential manner including 'Project Overview, Miscellaneous Tasks, Completed Tasks- Iteration 1, Product Backlog, Completed Tasks- Iteration 2, Handover & Showcase' & added members to each Trello card as per requirements. The juniors included important checklists related to each task in most Trello cards. **Bitbucket repository** was added to every team member in their local drive by forking & cloning. Every team member made commits to the Bitbucket repository after completing their designated work.

8.2. Definition of Done

Individual tasks are viewed as completed when they meet the following criteria:

1. The result of the task fulfills the prerequisites for the distinguished target deliverable it comes under.

2. All the relevant code should be without errors, clearly formatted & commented so that it can be easily comprehended by audience.

3. After completing the work, it should be committed to Bitbucket under senior supervision.

4. After committing to Bitbucket it should be marked in Trello cards in their respective sections.

Following this in a sequential manner, we as team managed to deliver our project deliverables efficiently in time. It improved the overall team building & pace of the work.

8.3. Task Review

The task review process of our project team has been very thorough following a particular hierarchy where juniors and seniors both contribute their work in a systematic way.

In our Studio sessions, internal scrum meeting & squad meetings we discussed the workflow of our project following project scoping including the list of project deliverables, insights, results & updates. Each member of our team then gets themselves enrolled in a task and update it to Trello card with proper checklist & due date. After debugging the codes & deliverables, the project deliverables are then assessed by seniors for correction & formatting.

In the project, every team member has access to Bitbucket repository and after completing their respective tasks, each member commits to Bitbucket under senior supervision while satisfying the above definition of 'Done' which is later presented to panellists, clients & mentor.

8.4. Testing

Throughout the Trimester, during Squad meetings, Studio Sessions & internal scrum meetings both seniors & juniors would cooperatively settle on which components we ought to include in Machine learning, Data Visualization & Web-Development. Further, we set up few sessions where we rigorously tested each task before showcasing to the audience. For the machine learning front, we used Jupyter & Google Colab and shared our results to the Bitbucket repository after debugging under senior supervision. Through which the team members evaluated each other's work on their respective project deliverables & also took references to include them in future iterations.

For the Data Visualization front, the team members gathered CSV files from the machine learning team after their task completion and built visualizations in Tableau to showcase to the panellists, clients & mentor. Afterwards, the visualization output & pickle file (from Machine Learning front) is sent to the Web-Development Team for integration. The Web-Development team handled the front end and backend where we Tested and fixed issues, enhanced UI | Homepage and Inner pages, created Dashboards for tableau and live machine learning models, provided utilities like subscription and contact us, added two factor authentication, integrated analysis from data visualization team, deployed Machine learning models & hosted the website on 'Heroku'. The overall testing process boosted the team building & technical skills which helped our team to achieve project deliverables in a given time frame.

8.5. Branching Strategy

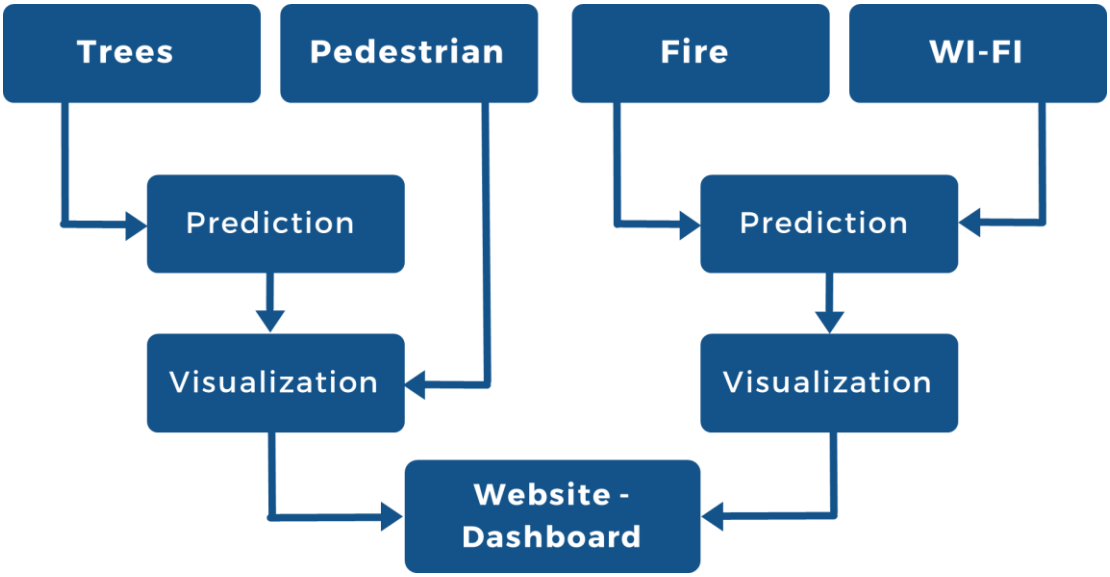
Throughout this Trimester, we have used a very convenient approach in managing the Bitbucket repository. As each team member worked on their own fork of the master branch by cloning and after getting their task assessed by the seniors, they committed the tasks on Bitbucket with detailed comments with the master branch & later created pull requests. Once completed, it was merged with the main deployable branch. Before committing any new task, each member was first told to pull any pending requests in order to avoid conflicts and errors.

This approach didn't create any conflicts between the commits & was easy to follow throughout the semester which paced our project & we were able to deliver exceptional results at the end.

9. Product Architecture

9.1. UML Diagram

Now, let us look at the UML diagram, where we see that the final product as a website comprising Prediction and Visualization components for each problem:



9.2. Tech Stack

When it comes to Tech Stack, we didn't have any dependencies from the client side. For Machine Learning, we have used 'Google Colab' & 'Jupyter' which gives GPU alternatives too. For Data Visualization, we have used 'Tableau Desktop' with 'Tableau Server' and for Web development, we have used 'HTML5, Bootstrap, Php, JavaScript, SASS and MySQL'.

10. Source Code

All source code for the tech stack of this project can be found in the project Bitbucket repositories below:

All data processing can be found in "**Datasets T32021**":
(https://bitbucket-students.deakin.edu.au/projects/D2IC-PG/repos/d2igeelong_ml/browse).

All machine learning Jupiter notebooks can be found "**ML-Code-T32021**":
(https://bitbucket-students.deakin.edu.au/projects/D2IC-PG/repos/d2igeelong_ml/browse).

All machine learning Jupiter notebooks can be found "**Visualisation – T32021**":
(https://bitbucket-students.deakin.edu.au/projects/D2IC-PG/repos/d2igeelong_ml/browse)

All **web development files** can be found in this repository:
(https://bitbucket-students.deakin.edu.au/projects/D2IC-PG/repos/d2igeelong_web/browse)

All source code should be found on your team's Deakin Bitbucket, unless your project has unique constraints that require you to store your code elsewhere. This includes any resources (e.g., wireframes, designs) that need to be transferred over to the new team as well.

Please provide all of the necessary instructions to accessing your source code. This includes URLs of online hosted repositories, links to any software dependencies, database components, or external libraries. If your code is hosted on a server external to Deakin, make sure to also transfer digital copies of your code over to your client and the next squad as a backup.

11. Login Credentials

For Fire Problem:

New login credentials will be created for a new member as per the role requirements, as they would be private.

For common data access:

Use the link below for NASA data access: https://urs.earthdata.nasa.gov/
(Free account sign up needed for access)

Use the link below for weather data access:
https://worldweatheronline.com

(Free 60-day account used for access of weather data. Any team member can sign up for a free 60-day account which comes with an API key).

For Tree, Pedestrian & Wi-Fi Problem:

The required Dataset can be acquired through the following link:
https://www.geelongdataexchange.com.au/pages/home/

Webpages source code is pushed to GitHub. The credential used for GitHub repository are:

Repository URL: https://github.com/D2i-GeelongCity?tab=repositories
Email: d2i.geelongcity@gmail.com
Username: D2i-GeelongCity
Password: haha@098@haha

12. Other Relevant Information

Training Resources:

- https://www.youtube.com/watch?v=F5mRW0jo-U4&t=6096s

  This is a 4-hour YouTube video by www.freecodecamp.org which talks about installation & configuration of Django. It also talks about working with forms, patterns, data in Django.

- https://docs.djangoproject.com/en/4.0/intro/tutorial01/

This space talk about how to write first application in Django with detail example. “Creation of a basic poll application” has been used as an example.

Deploying a website with Netlify from GitHub

Here is the simple step to deploy website on Netlify platform.

1. Push source code and files to GitHub.
2. Create a Netlify Account.
3. In "Overview" page select New site from Git
4. Authorize the I Git repository
5. Select the repo you want to deploy. Enter the build command and the folder for Netlify to distribute. For example, “parcel build index.html” and “dist” for deploying a static site with Parcel
6. Wait for app to deploy. In a few minutes, Netlify will give a URL to view the website. For more information: https://www.netlify.com/blog/2016/09/29/a-step-by-step-guide-deploying-on-netlify

13. Appendices

This is the link of the showcase video:

https://deakin365.sharepoint.com/:v:/r/sites/Data2IntelligenceConsulting/Shared%20Documents/D2I%20(Geelong%20City)/Trimester%203%20-%202021/D2i_GeelongCity_Showcase_video_T3_2021.MP4?csf=1&web=1&e=N7AHGd

The location of the showcase video (in MS Teams) is:
D2I (Geelong City) -> Trimester 3-2021 -> **D2i_GeelongCity_Showcase_video_T3**