Michael Christopher – s224830467

CAPRI - Correlation of Air Pollution and Respiratory Illness

What have been done?

1. **Finalised Data Collection for Air Quality Index**

   Completed the acquisition of AQI datasets across multiple regions, ensuring coverage for core pollutants (CO, NO_2, O_3, PM10, PM2.5).

   So what? This establishes a solid foundation for analysis, ensuring our models are supported by comprehensive and reliable inputs.

Data from https://discover.data.vic.gov.au/dataset/epa-air-watch-all-sites-air-quality-hourly-averages-yearly

	A	B	C	D	E	F	G	H	I	J	K	L
1	datetime_AEST	datetime_local	location_id	location_n	DBT	NO2	Sigma60	SO2	SWD	SWS	VWD	VWS
2	2023-12-31 23:00:00	2024-01-01 00:00:00	10006	Altona North	16,498	0,966	14,93	0,611	154,087	2,894	153,371	2,802
3	2024-01-01 00:00:00	2024-01-01 01:00:00	10006	Altona North	16,491		15,488		152,279	2,952	151,498	2,851
4	2024-01-01 01:00:00	2024-01-01 02:00:00	10006	Altona North	16,503	1,441	14,504	1,033	146,637	2,465	146,034	2,395
5	2024-01-01 02:00:00	2024-01-01 03:00:00	10006	Altona North	16,122	2,314	14,623	1,295	122,633	2,469	122,036	2,396
6	2024-01-01 03:00:00	2024-01-01 04:00:00	10006	Altona North	15,427	2,78	16,766	2,242	112,545	2,08	113,91	2,011
7	2024-01-01 04:00:00	2024-01-01 05:00:00	10006	Altona North	14,554	3,037	21,651	2,032	97,454	1,215	99,707	1,148
8	2024-01-01 05:00:00	2024-01-01 06:00:00	10006	Altona North	14,1	6,428	84,389	1,277	154,11	0,616	46,532	0,058
9	2024-01-01 06:00:00	2024-01-01 07:00:00	10006	Altona North	17,264	5,117	43,5	2,552	193,91	1,189	179,388	0,982
10	2024-01-01 07:00:00	2024-01-01 08:00:00	10006	Altona North	18,794	3,032	30,321	2,529	159,401	1,644	155,779	1,48
11	2024-01-01 08:00:00	2024-01-01 09:00:00	10006	Altona North	19,842	2,453	35,143	3,116	153,118	1,899	150,671	1,644
12	2024-01-01 09:00:00	2024-01-01 10:00:00	10006	Altona North	21,853	2,16	20,956	3,25	159,511	2,375	159,362	2,248
13	2024-01-01 10:00:00	2024-01-01 11:00:00	10006	Altona North	22,854	1,806	21,844	2,709	164,611	2,728	163,222	2,572
14	2024-01-01 11:00:00	2024-01-01 12:00:00	10006	Altona North	24,282	2,093	20,291	2,222	169,128	3,235	167,619	3,066
15	2024-01-01 12:00:00	2024-01-01 13:00:00	10006	Altona North	25,112	1,483	19,474	2,248	175,013	3,636	174,064	3,467
16	2024-01-01 13:00:00	2024-01-01 14:00:00	10006	Altona North	25,134	1,422	19,777	2,201	180,367	4,034	179,189	3,838
17	2024-01-01 14:00:00	2024-01-01 15:00:00	10006	Altona North	24,812	1,329	20,945	1,765	186,204	4,292	184,903	4,067
18	2024-01-01 15:00:00	2024-01-01 16:00:00	10006	Altona North	24,251	1,167	18,864	1,542	180,814	4,258	179,587	4,082
19	2024-01-01 16:00:00	2024-01-01 17:00:00	10006	Altona North	24,765	2,598	19,277	1,355	183,494	4,019	181,829	3,832
20	2024-01-01 17:00:00	2024-01-01 18:00:00	10006	Altona North	23,048	2,22	22,077	1,748	195,898	3,719	193,946	3,503
21	2024-01-01 18:00:00	2024-01-01 19:00:00	10006	Altona North	20,683	1,029	24,47	1,676	198,82	3,057	198,851	2,86
22	2024-01-01 19:00:00	2024-01-01 20:00:00	10006	Altona North	19,409	1,241	23,773	1,38	200,62	2,418	199,187	2,261
23	2024-01-01 20:00:00	2024-01-01 21:00:00	10006	Altona North	18,971	1,12	23,916	1,179	199,015	1,931	198,247	1,8
24	2024-01-01 21:00:00	2024-01-01 22:00:00	10006	Altona North	18,622	1,307	25,925	1,244	207,175	1,568	208,058	1,44
25	2024-01-01 22:00:00	2024-01-01 23:00:00	10006	Altona North	18,365	0,893	26,402	0,306	207,955	1,567	207,828	1,438
26	2024-01-01 23:00:00	2024-01-02 00:00:00	10006	Altona North	18,531	1,115	29,142	-0,555	210,897	1,381	213,433	1,252
27	2024-01-02 00:00:00	2024-01-02 01:00:00	10006	Altona North	18,862		28,272		217,359	1,082	220,006	0,993

Data from Air Pollution in Victoria: Real-time Air Quality Index Visual Map

	A	B	C	D
1	date, pm25, pm10, o3, no2, so2, co			
2	2025/8/1, 43, 13, 13, 13, , 3			
3	2025/8/2, 41, 20, 16, 12, , 4			
4	2025/8/3, 73, 11, 22, 8, , 1			
5	2025/8/4, 31, 12, 22, 6, ,			
6	2025/8/5, 22, 14, 18, 10, , 1			
7	2025/8/6, 21, 15, 20, 15, , 2			
8	2025/8/7, 32, 16, 16, 15, , 3			
9	2025/8/8, 40, 20, 13, 13, , 4			
10	2025/8/9, 60, 21, 19, 11, , 4			
11	2025/8/10, 68, 18, 19, 12, , 3			
12	2025/8/11, 56, 11, 15, 8, , 1			
13	2025/8/12, 32, 15, 20, 13, , 2			
14	2025/8/15, 34, 10, 23, 10, , 1			
15	2025/8/16, 33, 10, 19, 8, , 1			
16	2025/8/17, 31, 12, 24, 10, , 1			
17	2025/8/18, 28, 14, 1, 12, , 4			
18	2025/8/19, 49, , , , ,			
19	2025/7/1, 36, 9, 18, 9, , 1			
20	2025/7/2, 18, 16, 14, 14, , 3			
21	2025/7/3, 36, 15, 14, 9, , 3			
22	2025/7/4, 43, 10, 18, 7, , 2			
23	2025/7/5, 31, 17, 10, 14, , 4			
24	2025/7/6, 49, 18, 14, 10, , 4			
25	2025/7/7, 57, 8, 19, 4, , 1			
26	2025/7/8, 20, 10, 26, 3, , 2			

2. **Developed Data Cleaning & Preprocessing Script**

   Implemented Python scripts to handle missing values, align timestamps, and standardise pollutant units across datasets.

   So what? Automated preprocessing streamlines future updates, reduces error risk, and ensures consistent quality in downstream modelling.

```
# Read a sheet by its name
df_morwell_east = pd.read_excel('datasets/2024_All_sites_air_quality_hourly_avg_AIR-I-F-V-VH-O-S1-DB-M2-4-0.xlsx', sheet_name='Morwell East_10017')
df_mooroolbark = pd.read_excel('datasets/2024_All_sites_air_quality_hourly_avg_AIR-I-F-V-VH-O-S1-DB-M2-4-0.xlsx', sheet_name='Mooroolbark_10136')
df_altona_north = pd.read_excel('datasets/2024_All_sites_air_quality_hourly_avg_AIR-I-F-V-VH-O-S1-DB-M2-4-0.xlsx', sheet_name='Altona North_10006')
df_dandenong = pd.read_excel('datasets/2024_All_sites_air_quality_hourly_avg_AIR-I-F-V-VH-O-S1-DB-M2-4-0.xlsx', sheet_name='Dandenong_10022')
df_point_cook = pd.read_excel('datasets/2024_All_sites_air_quality_hourly_avg_AIR-I-F-V-VH-O-S1-DB-M2-4-0.xlsx', sheet_name='Point Cook_10005')
df_geelong_south = pd.read_excel('datasets/2024_All_sites_air_quality_hourly_avg_AIR-I-F-V-VH-O-S1-DB-M2-4-0.xlsx', sheet_name='Geelong South_10107')
df_traralgon = pd.read_excel('datasets/2024_All_sites_air_quality_hourly_avg_AIR-I-F-V-VH-O-S1-DB-M2-4-0.xlsx', sheet_name='Traralgon_10011')
df_alphington = pd.read_excel('datasets/2024_All_sites_air_quality_hourly_avg_AIR-I-F-V-VH-O-S1-DB-M2-4-0.xlsx', sheet_name='Alphington_10001')
df_melton = pd.read_excel('datasets/2024_All_sites_air_quality_hourly_avg_AIR-I-F-V-VH-O-S1-DB-M2-4-0.xlsx', sheet_name='Melton_10169')
df_melbourne_cbd = pd.read_excel('datasets/2024_All_sites_air_quality_hourly_avg_AIR-I-F-V-VH-O-S1-DB-M2-4-0.xlsx', sheet_name='Melbourne CBD_10239')
df_box_hill = pd.read_excel('datasets/2024_All_sites_air_quality_hourly_avg_AIR-I-F-V-VH-O-S1-DB-M2-4-0.xlsx', sheet_name='Box Hill_10042')
df_brighton = pd.read_excel('datasets/2024_All_sites_air_quality_hourly_avg_AIR-I-F-V-VH-O-S1-DB-M2-4-0.xlsx', sheet_name='Brighton_10007')
df_morwell_south = pd.read_excel('datasets/2024_All_sites_air_quality_hourly_avg_AIR-I-F-V-VH-O-S1-DB-M2-4-0.xlsx', sheet_name='Morwell South_10217')
df_macleod = pd.read_excel('datasets/2024_All_sites_air_quality_hourly_avg_AIR-I-F-V-VH-O-S1-DB-M2-4-0.xlsx', sheet_name='Macleod_10377')
df_brooklyn = pd.read_excel('datasets/2024_All_sites_air_quality_hourly_avg_AIR-I-F-V-VH-O-S1-DB-M2-4-0.xlsx', sheet_name='Brooklyn_1112')
```

In this case, I use a formula in https://document.airnow.gov/technical-assistance-document-for-the-reporting-of-daily-air-quailty.pdf to calculate the AQI

Calculating the AQI from pollutant concentration data

The AQI is the highest value calculated for each pollutant as follows:

1. Identify the highest concentration among all of the monitors within each reporting area and truncate as follows:

   Ozone (ppm) – truncate to 3 decimal places

   $PM_{2.5}$ ($\mu g/m^3$) – truncate to 1 decimal place

   PM_{10} ($\mu g/m^3$) – truncate to integer

   CO (ppm) – truncate to 1 decimal place

   SO_2 (ppb) – truncate to integer

   NO_2 (ppb) – truncate to integer

2. Using Table 6 (next page), find the two breakpoints that contain the concentration.

3. Using Equation 1, calculate the index.

4. Round the index to the nearest integer.

Equation 1:

$$I_p = \frac{I_{Hi} - I_{Lo}}{BP_{HI} - BP_{Lo}}(C_p - BP_{Lo}) + I_{Lo}.$$

Where I_p = the index for pollutant p

C_p = the truncated concentration of pollutant p

BP_{Hi} = the concentration breakpoint that is greater than or equal to C_p

BP_{Lo} = the concentration breakpoint that is less than or equal to C_p

I_{Hi} = the AQI value corresponding to BP_{Hi}

I_{Lo} = the AQI value corresponding to BP_{Lo}

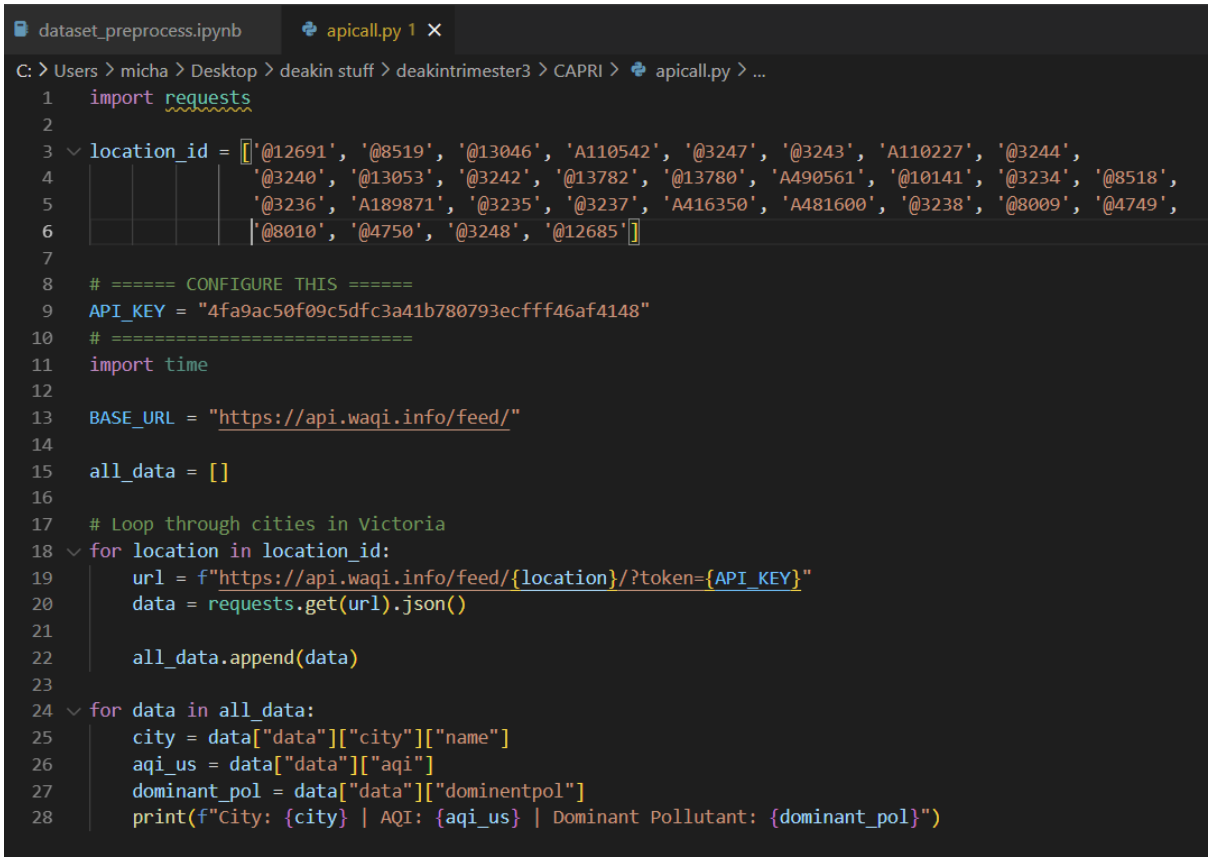
These Breakpoints...							...equal this AQI	...and this category
O_3 (ppm) 8-hour	O_3 (ppm) 1-hour[1]	$PM_{2.5}$ (µg/m³) 24-hour	PM_{10} (µg/m³) 24-hour	CO (ppm) 8-hour	SO_2 (ppb) 1-hour	NO_2 (ppb) 1-hour	AQI	
0.000 - 0.054	-	0.0 – 9.0	0 - 54	0.0 - 4.4	0 - 35	0 - 53	0 - 50	Good
0.055 - 0.070	-	9.1 – 35.4	55 - 154	4.5 - 9.4	36 - 75	54 - 100	51 - 100	Moderate
0.071 - 0.085	0.125 - 0.164	35.5 – 55.4	155 - 254	9.5 - 12.4	76 - 185	101 - 360	101 - 150	Unhealthy for Sensitive Groups
0.086 - 0.105	0.165 - 0.204	(55.5 - 125.4)[3]	255 - 354	12.5 - 15.4	[3] 186 - 304	361 - 649	151 - 200	Unhealthy
0.106 - 0.200	0.205 - 0.404	(125.5 - (225.4)[3]	355 - 424	15.5 - 30.4	[3] 305 - 604)	650 - 1249	201 - 300	Very unhealthy
0.201-([2])	0.405+	225.5+	425+	30.5+	[3] 605+	1250+	301+	Hazardous [4]

```
pollutant_columns = ['O3', 'CO', 'SBPM25', 'PM10', 'NO2', 'SO2']
cleaned_columns = ['datetime_local'] + pollutant_columns

def removeUnwantedColumns(df):
    df_columns = []
    for column in df.columns:
        if column in cleaned_columns:
            df_columns.append(column)
    return df[df_columns]

df_morwell_east_removed = removeUnwantedColumns(df_morwell_east)
df_mooroolbark_removed = removeUnwantedColumns(df_mooroolbark)
df_altona_north_removed = removeUnwantedColumns(df_altona_north)
df_dandenong_removed = removeUnwantedColumns(df_dandenong)
df_point_cook_removed = removeUnwantedColumns(df_point_cook)
df_geelong_south_removed = removeUnwantedColumns(df_geelong_south)
df_traralgon_removed = removeUnwantedColumns(df_traralgon)
df_alphington_removed = removeUnwantedColumns(df_alphington)
df_melton_removed = removeUnwantedColumns(df_melton)
df_melbourne_cbd_removed = removeUnwantedColumns(df_melbourne_cbd)
df_box_hill_removed = removeUnwantedColumns(df_box_hill)
df_brighton_removed = removeUnwantedColumns(df_brighton)
df_morwell_south_removed = removeUnwantedColumns(df_morwell_south)
df_macleod_removed = removeUnwantedColumns(df_macleod)
df_brooklyn_removed = removeUnwantedColumns(df_brooklyn)
```

3. **Tested API Call with WAQI (World Air Quality Index) API**

Successfully wrote and executed test code to fetch real-time AQI data, validating the feasibility of API-based integration.

So what? This capability enables dynamic data ingestion, supporting future extensions like interactive dashboards or predictive monitoring.



4. **Air Quality Prediction Models**

Began predictive modelling using multiple machine learning approaches (e.g., Linear Regression, Random Forest, Gradient Boosting, etc.) to forecast AQI values, aiming to identify the model with the best predictive performance.

```python
X = df[features]
y = df[target]

# 5. Split Data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, shuffle=False)

# 6. Scale Features
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# 7. Train and Compare Models
models = {
    'LinearRegression': LinearRegression(),
    'RandomForest': RandomForestRegressor(random_state=42),
    'GradientBoosting': GradientBoostingRegressor(random_state=42)
}

for name, model in models.items():
    model.fit(X_train_scaled, y_train)
    y_pred = model.predict(X_test_scaled)
    mse = mean_squared_error(y_test, y_pred)
    r2 = r2_score(y_test, y_pred)
    print(f"{name} - MSE: {mse:.2f}, R2: {r2:.2f}")

# 8. Predict Future pm25 with Each Model
last_row = df.iloc[-1]
future_features = [last_row[f'pm25_lag{lag}'] for lag in range(1, 4)] + [last_row[' pm10'], last_row['
future_features_scaled = scaler.transform([future_features])
for name, model in models.items():
    future_pm25 = model.predict(future_features_scaled)
    print(f"{name} predicted future pm25: {future_pm25[0]:.2f}")
```

```
       date  pm25  pm10   o3  no2  so2  co
0  2025/8/1    27    13   13   13        3
1  2025/8/2    32    20   16   12        4
2  2025/8/3    51    11   22    8        1
3  2025/8/4    18    12   22    6
4  2025/8/5     8    14   18   10        1
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3859 entries, 0 to 3858
Data columns (total 7 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   date    3859 non-null   object
 1    pm25   3859 non-null   object
 2    pm10   3859 non-null   object
 3    o3     3859 non-null   object
 4    no2    3859 non-null   object
 5    so2    3859 non-null   object
 6    co     3859 non-null   object
dtypes: object(7)
memory usage: 211.2+ KB
None
LinearRegression - MSE: 62.46, R2: 0.13
RandomForest - MSE: 59.23, R2: 0.17
RandomForest - MSE: 59.23, R2: 0.17
GradientBoosting - MSE: 57.62, R2: 0.19
LinearRegression predicted future pm25: 21.20
RandomForest predicted future pm25: 15.77
```

Next actions?

1. Refine preprocessing pipeline for full dataset integration with health data (AIHW).
2. Begin exploratory data analysis (EDA) to identify patterns and validate dataset reliability.
3. Prototype initial visualisations (e.g., pollutant trends, heatmaps) for technical and non-technical audiences.
4. Work more on the machine learning modelling so it is more in depth and meaningful.