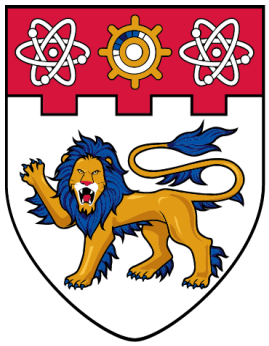


NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

SC4020 Data Analytics and Mining

Project 2

GROUP 32

KIM CHAIYOUNG (U2240797L)

LEE SANGHYUN (U2020666G)

SEW KAI TEK (U2240436H)

XIAO LINGYI (U2020057G)

1. Analysis of Co-occurrence Patterns of Points of Interest (POI)

1.1. Data Preparation

Firstly, data was prepared such that each grid cell represents a basket, with POI categories appearing in the grid as items. The required libraries, POI data for the four cities and the POI categories' data were loaded. The category IDs in the POI categories were manually added to merge the data so that every grid cell in each city's data has both category ID and name.

1.2. Creating Baskets

Next, baskets were created in the dictionary form {Category: Quantity}. The data frame was grouped by unique (x,y) coordinates, with each group subsequently transformed into a dictionary by the lambda function, resulting in a dictionary where each POI category (Category) for the grid cells was mapped to its count (Quantity). **Table 1** below shows an example of results, displaying the first 5 results of city A's basket data for each x,y coordinate pair.

	x	y	basket
0	1	1	{'Church': 4, 'Transit Station': 4, 'Building Material': 2, 'Hair Salon': 2, 'Community Center': 1, 'Real Estate': 1, 'Kindergarten': 1}
1	1	2	{'Post Office': 2, 'Gardening': 1, 'Transit Station': 1, 'Accountant Office': 1, 'Home Appliances': 1}
2	1	3	{'Church': 4, 'Heavy Industry': 2, 'Transit Station': 2, 'NPO': 2, 'Elderly Care Home': 1, 'School': 1, 'Kindergarten': 1, 'Community Center': 1, 'Convenience Store': 1, 'Home Appliances': 1}
3	1	4	{'Community Center': 3, 'Heavy Industry': 2, 'Driving School': 1, 'Transit Station': 1, 'Building Material': 1, 'Church': 1, 'Hair Salon': 1}
4	1	5	{'Church': 2, 'Home Appliances': 1, 'Transit Station': 1, 'Hair Salon': 1, 'Laundry ': 1, 'Accountant Office': 1, 'Building Material': 1}

Table 1. Data table with city A's basket data for each x,y coordinate pair, showing first 5 results

1.3. One-hot encoding

To begin, one-hot encoding was conducted to generate transaction arrays to have an array format suitable for implementing the Apriori algorithm (Aditya, 2023). Transactions for each of the four cities were prepared with the list of POI categories in each grid cell. Each basket represents the POI categories that appear together in one grid cell, enabling each grid cell to become a "transaction". Subsequently, all transactions from four cities were combined to ensure that each city had consistent columns in the one-hot encoded data frames later when running the algorithm. Transactions were transformed into a binary array using the TransactionEncoder() function, and the fit method was used to learn the set of POI categories across all transactions. Afterwards, the transform method was applied to each city's transaction to convert each city's POI categories into a binary array. Each cell contained '1' if the POI category was present in the particular grid cell, and '0' if absent, with each city's transformed array being converted to a data frame.

1.4. Generating Frequent Itemsets using apriori() function

Frequent itemsets were obtained for each city using the apriori() function with minimum support thresholds between 0.1 and 0.4. For each city, higher minimum support thresholds (above the chosen range) resulted in no frequent itemsets, meaning no significant POI combinations were found with those higher thresholds. The co-occurrence matrix was visualised using heatmaps to

highlight the patterns and relationships between different POIs in each city. For each city, different minimum support thresholds were chosen as minimum support thresholds higher than the chosen values resulted in no itemsets with co-occurring places. **Table 2** below shows the co-occurrence heatmaps for each city, the top 5 most frequent itemsets combinations and the main points about the top combinations.

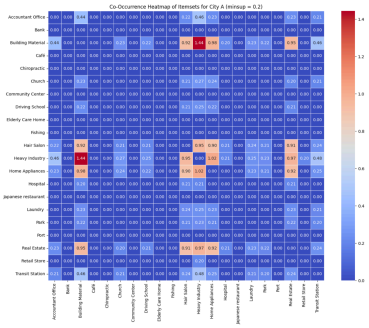
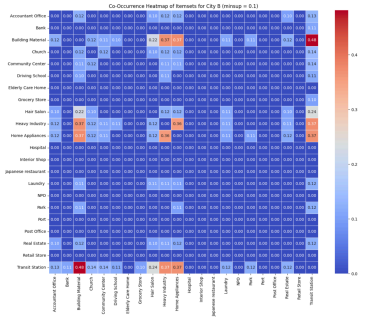
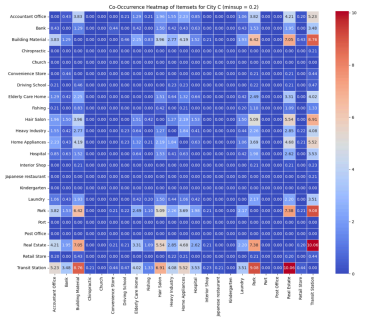
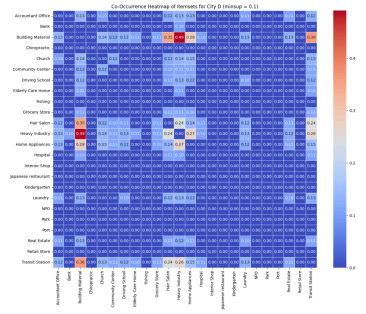
	City A (minsup = 0.2) 1.44: Heavy Industry & Building Material 0.98: Home Appliances & Building Material 0.97: Heavy Industry & Real Estate 0.95: Real Estate & Building Material 0.92: Hair Salon & Building Material → *Strong, highly interlinked sectors like heavy industry, real estate, and building materials dominate the city's landscape.*
	City B (minsup = 0.1) 0.48: Transit Station & Building Material 0.37: Heavy Industry & Building Material 0.37: Home Appliances & Building Material 0.24: Hair Salon & Transit Station 0.22: Hair Salon & Building Material → *Weaker sector relationships, with a focus on building material and transit stations*
	City C (minsup = 0.2) 10.06: Real Estate & Transit Station 9.08: Park & Transit Station 8.76: Building Material & Transit Station 7.38: Park & Real Estate 7.05: Building Material & Real Estate → *Extremely strong connections, especially between real estate, transit stations, and parks, highlight urban development and infrastructure.*
	City D (minsup = 0.1) 0.49: Heavy Industry & Building Material 0.36: Transit Station & Building Material 0.35: Hair Salon & Building Material 0.28: Home Appliances & Building Material 0.27: Home Appliances & Heavy Industry → *Weaker sector overlap, especially in heavy industry and building material*

Table 2: Co-occurrence heatmaps for each city and top 5 most frequent itemsets combinations

2. Mining Sequential Patterns

2.1. Data Preparation

To generate and perform “tripleg” clearly, the dataset was refined by removing rows with missing coordinates and combining the date and time columns into a single datetime column. Additionally, due to the large scale of the data, the starting date was set as 2023-01-01 for better computational processes. The dataset was further modified by connecting with POI data for each city, mapping the category IDs to human readable names such as “Restaurant” and “Park”, and assigning “Unknown” to any invalid category IDs. The dataframe was converted into GeoDataframe which performs geospatial operations, and was assigned a CRS value of EPSG:3857. Given that the dataset provided uses 500 metre grid format and that EPSG:4326 was better compatible with latitude-longitude coordinates, and not with the grid-based coordinates, EPSG:3857 was ultimately chosen. (Auravant Developers, 2022)

2.2. Creating Triplegs

The process of developing tripleg makes handling a large dataset inevitable. Chunk-based data processing was applied, enabling incremental loading with an overloading dataset and warranting precision and flexibility of the large datasets. The process_city_in_chunks function was built to iteratively process the data within developed chunk data for each city’s dataset. This function begins with replacing invalid coordinates such as ‘-999’ into ‘NaN’ and filtering only the first 30 days to concentrate on consistent temporal timestamps. Next, the dataset was converted into a geospatial format for efficient advanced spatial operations. Through the generate_staypoints function in trackintel library, the staypoints were generated to capture significant stationary locations. While trackintel only supports haversine distance metric which makes it unsuitable for grid-based coordinates, this limitation will be mitigated without any modification or conversion of data as the task focuses on grid cell transitions rather than precise geographic distances. Finally, each tripleg was associated with the nearest POI by calculating spatial distance. This methodology of creating tripleg offered better memory efficiency and geospatial analysis for precise and flexible insights in mobility and POI association analysis.

2.3. Data Analysis

After creating triplegs for each city, the GSP algorithm was implemented and various types of graphs were constructed. Firstly, the spatial distribution of POI across the four cities based on their geographical coordinates and categorical labels are shown below:

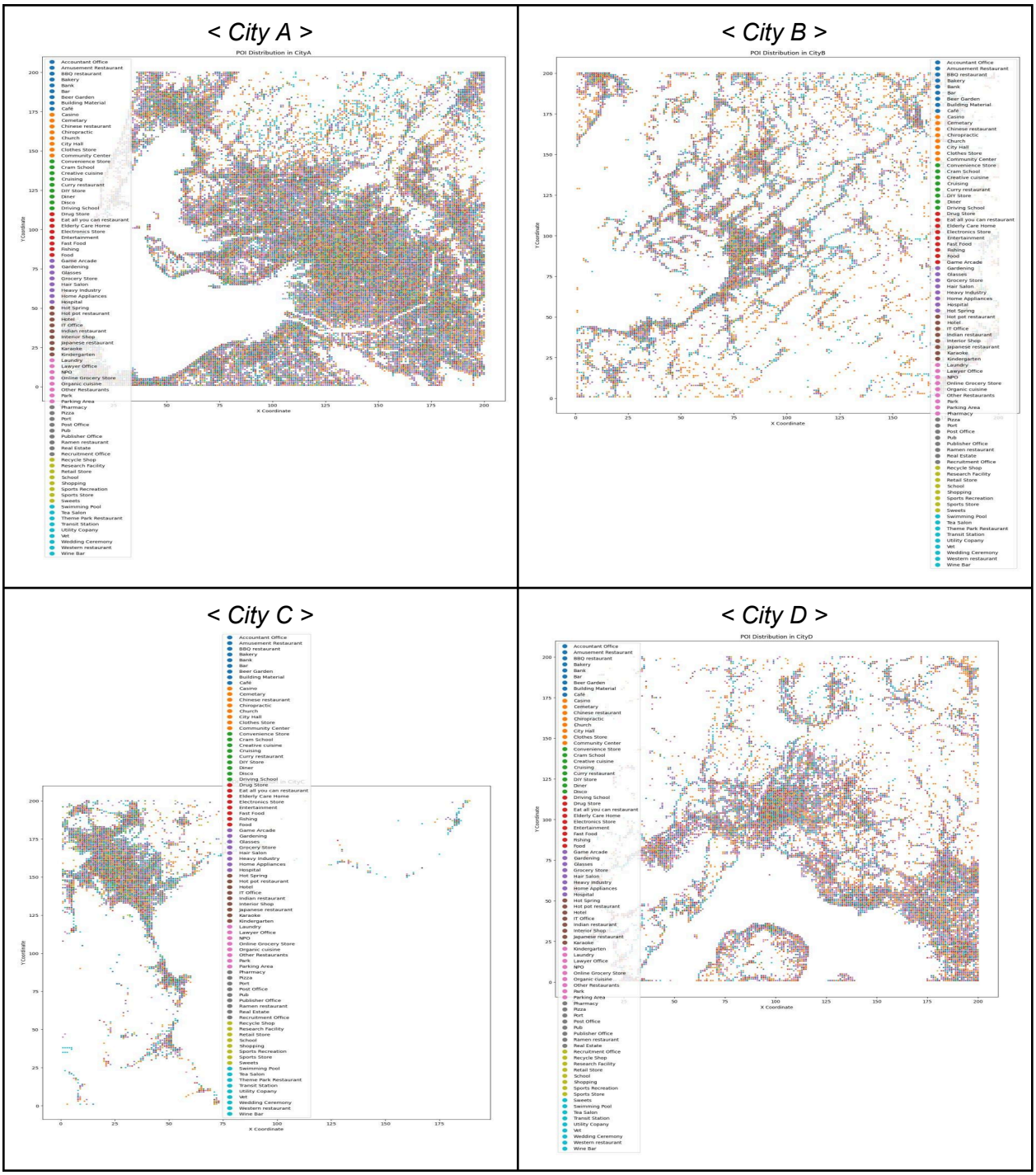


Table 3: POI Distribution Graph for City A, B, C and D

From **Table 3**, the clusters, distributions, and density of the POI category for each city can be easily distinguished. For example, City A tends to display a dense clustering pattern in the central region. Additionally, categories which indicate residential places such as “Restaurants”, and “Community Centers” dominate the clusters. Through the analysis, city A can be assumed to be a well-designed urban area. Apart from analysing within individual cities, comparisons between cities can be made. Unlike City A, Cities B and D display less concentrated, more

dispersed patterns, implying suburban areas. Moreover, City A and C had a greater diversity of POI categories compared to City B and D. This representation of data analysis visualises the density and distribution of POI data, allowing abundant insights at a glance.

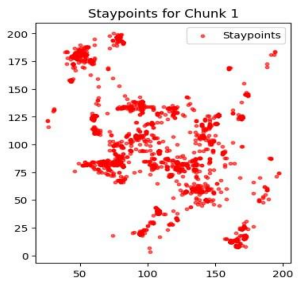
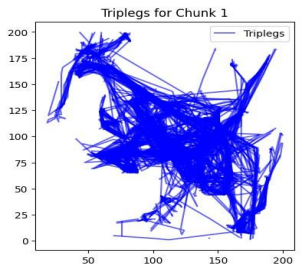
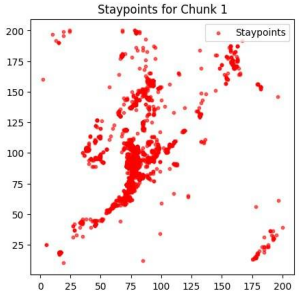
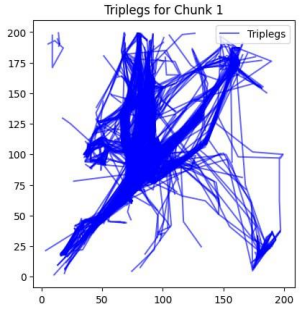
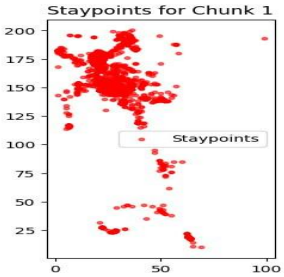
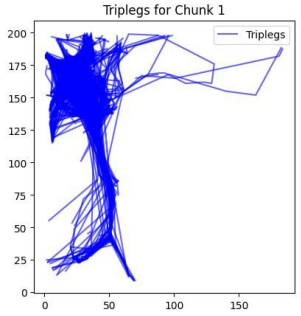
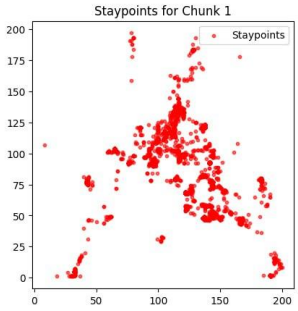
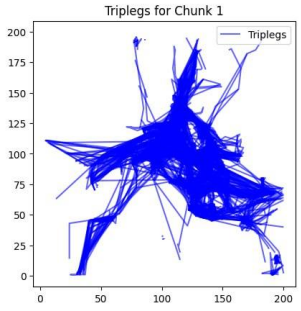
	Staypoints	Triplegs
City A		
City B		
City C		
City D		

Table 4: DBSCAN Graph for City A, B, C and D

From **Table 4**, the staypoints offer a more consolidated and summarised view of the specific clusters of individuals. Moreover, the triplegs shed information on available routes for travel, inferring that the majority of the routes were straight lines that directly connect two stay points, suggesting that the traffic route is efficient (shortest possible distance) and cuts through rural areas (low population density).

Additionally, the histogram **(Figure 1)** below displays the top 10 most frequent single-place visits of City A. The “Unknown” parameter in the graph represents the certain coordinates that cannot be mapped back to POI data. Likewise, “Retail Store” and “Gardening” are the most visited places. These locations are identified as significant by the sequential mining algorithm because they appear frequently in the dataset based on the support threshold. Furthermore, the second visualisation **(Figure 2)** represents the sequential movement patterns which illustrate how individuals frequently visit in one sequence of trips. POIs that were related to each other by the age group of residents tend to be grouped together. For example, “Elderly Care Home” and “Gardening” are located next to each other, suggesting that elderly individuals are inclined to take up gardening as a hobby. Both additional analyses provide comprehensive perspectives of both stationary and dynamic movement patterns, shedding light on individuals’ preferences and the demographics of the city. Finally, the DBSCAN clustering of City B effectively classifies spatial points into clusters, as shown in **Figure 3**. Visualising clusters by using DBSCAN provides better insights into highly visited areas.

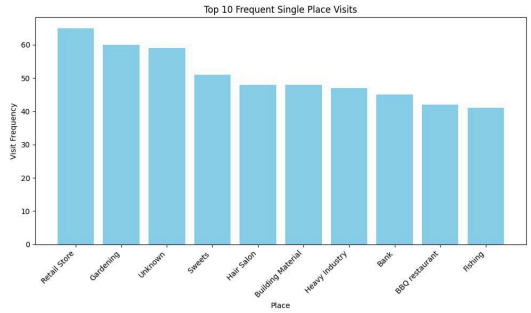


Figure 1: Histogram of Top 10 Single-Point Visits for City A

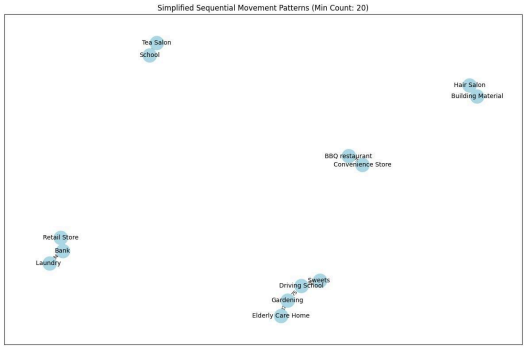


Figure 2: Simplified Sequential Movement Patterns of City A

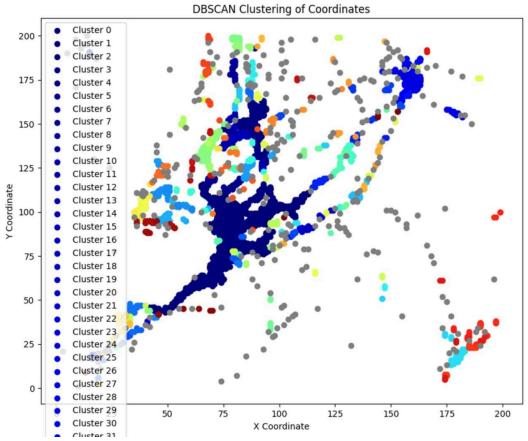


Figure 3: DBSCAN Clustering of City B

3. Open Advanced Tasks

For this open task, temporal patterns were generated to analyse the patterns in the number of unique visitors to various POIs. Two algorithms were utilised - ARIMA (Autoregressive Integrated Moving Average) and Prophet for future predictions.

3.1 Data Processing:

Firstly, raw POI data was processed by mapping it to POI category information and converted into geospatial format. Next, aggregation of the processed POI data for storage in the GeoDataFrame format for later use was done in manageable chunks for memory efficiency similar to task 2. The data was filtered to process data only for the first 21 days. Afterwards, the data was restructured to show weekly trends through temporal aggregation. The weekly trend data was used to calculate the total number of visitors for each POI category in each city and was normalised using Min-Max scaling to objectively compare trends across categories.

3.2 Analysis of Temporal Patterns

For each city, the percentage share was computed based on each POI category's visitors compared to the total visitors in a city for each week. **Table 5E** visualises the percentage shares of the "Church" category. The high percentage share of City D implies frequent visits to this place, suggesting more believers and more local people than tourists. Conversely, City C had the lowest percentage share for churches, which corresponds to their spikes in leisure and tourism-related categories as discovered from the codes. For a more selective analysis, specific key categories: "Transit Station", "Heavy Industry" and "Building Material", were chosen for computation and visualisation. **Table 5A** and **Table 5D** display the domination of industrial categories like "Heavy Industry" and "Transit Station" for City A and City D. **Table 5C** presents "Transit Station" as the highest category, which correlates with the high tourism categories for City C.

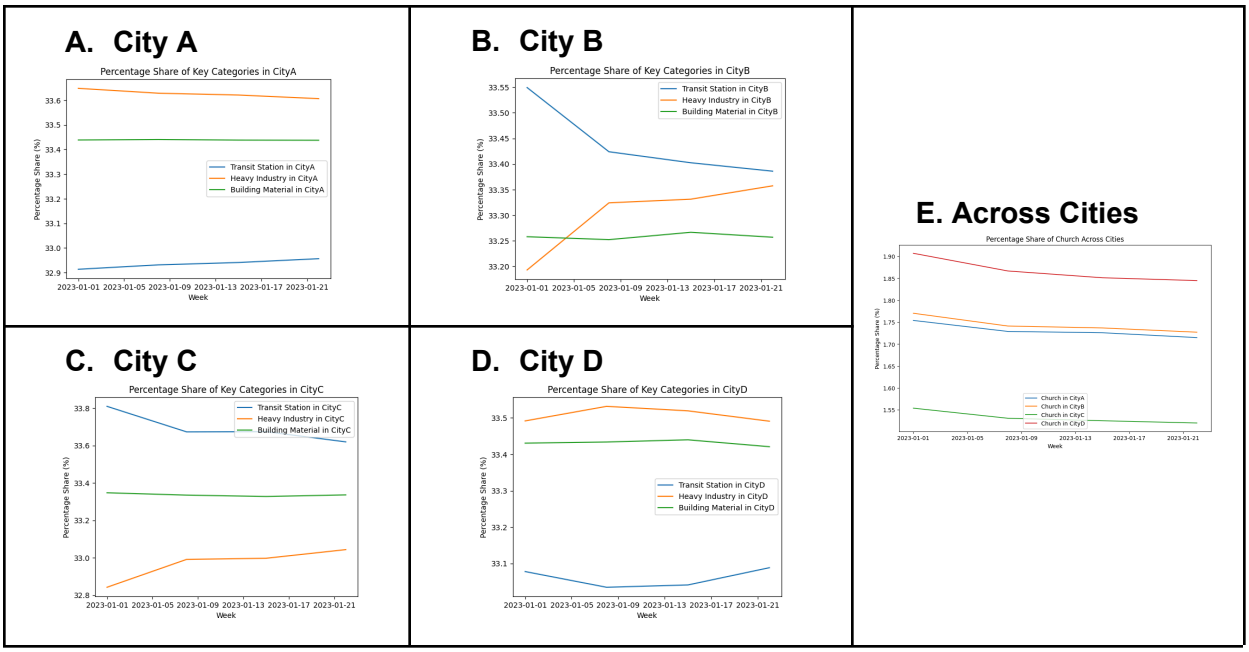


Table 5: Percentage Share Graphs for Key Categories and 'Church' Category

3.3 Utilising ARIMA for Future Trend Predictions

For further analysis, the ARIMA (AutoRegressive Integrated Moving Average) model was used for time-series forecasting. This was chosen as it is a feasible algorithm that captures trends, seasonality and autocorrelation, and is suitable for datasets that have regular intervals (Bektemyssova et al., 2022). After preprocessing the categorical data, the visitor counts were extracted as time-series data for ARIMA modelling. Subsequently, the ARIMA model was fitted with specific parameters and the next 10 days of future data was predicted. For visualisation, ARIMA's unique visitors' forecast for 'Heavy Industry' in City D is shown in **Table 6A**, being compared to the actual data of unique visitors.

3.4 Utilising Prophet Forecasting for Future Trend Predictions

To complement the ARIMA model, Prophet - a forecasting tool by Facebook, was also applied (Žunić et al., 2020). This tool impressively handles missing data, seasonality and trends like ARIMA. Similar data processing and model fitting steps as ARIMA were taken. Then, the Future DataFrame was created, and predictions were computed with daily frequencies for the predictions. The graph in **Table 6B** represents the Prophet's unique visitors' forecast for 'Accountant Office' category in City A. Prophet can be further used to visualise decomposed components: trend and seasonality, providing more opportunities for pattern analysis. For example, **Table 6C** and **Table 6D** trends in the plots reveal that the visits to 'Accountant Office' category are stable during the weekdays, but drop significantly on the weekends.

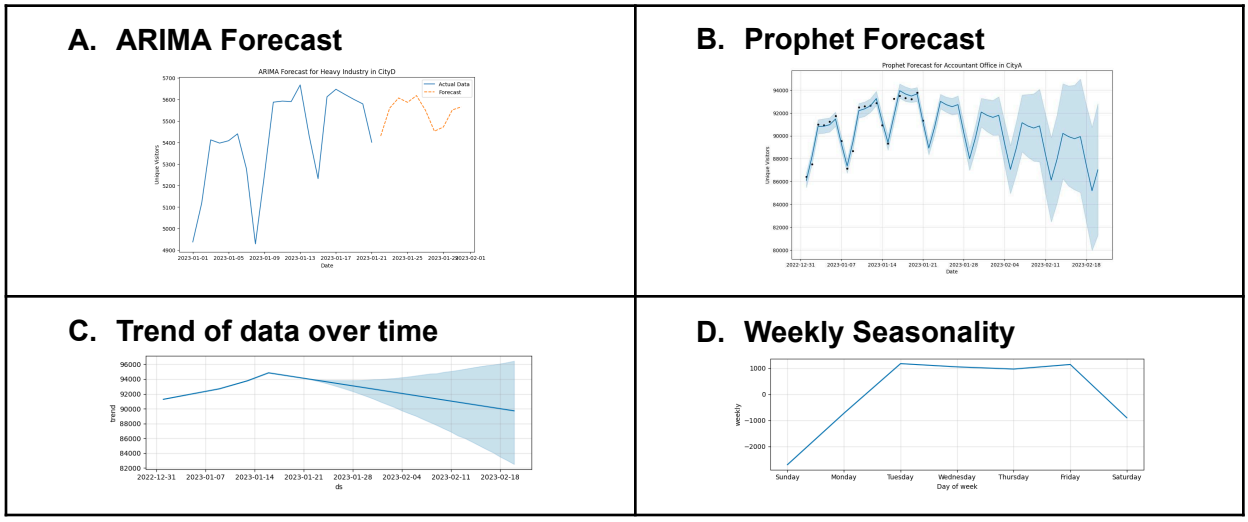


Table 6. Visualisations from ARIMA and Prophet forecasting results

3.5 Social and Business Applications

The analyses above on the patterns of mobility data with forecasting tools have significant social and business applications. These tools can be useful in applications such as understanding trends in visitor traffic for infrastructure and urban planning, location or mobility-based business strategies, understanding weekly patterns for event planning or even for understanding social behaviour through the visitor patterns in different regions. Ultimately, this work can help make data-driven decisions in many potential fields.

References

Aditya. (2023, May 20). *Implement apriori algorithm in Python*. Coding Infinite. https://codinginfinite.com/implement-apriori-algorithm-in-python/#google_vignette

Auravant Developers. (2022, September 9). *Understanding clustering algorithms for spatial data analysis*. https://developers.auravant.com/en/blog/2022/09/09/post-3/

Bektemyssova, G., Ahmad, A. R., Mirzakulova, S., & Ibraeva, Z. (2022). TIME SERIES FORECASTING BY THE ARIMA METHOD. *Scientific Journal of Astana IT University, 11,* 14–23. https://doi.org/10.37943/hfch4395

Žunić, E., Korjenić, K., Hodžić, K., & Đonko, D. (2020). Application of Facebook's Prophet algorithm for successful sales forecasting based on real-world data. *International Journal of Computer Science and Information Technology, 12*(2), 23–36. https://doi.org/10.5121/ijcsit.2020.12203

Individual Contribution Claims

KIM CHAIYOUNG (U2240797L): Task 1 Code, Task 1 Report Writing, Task 3 Report Writing

LEE SANGHYUN (U2020666G): Task 2 Code, Task 2 Report Writing

SEW KAI TEK (U2240436H): Task 2 Report Writing, Task 3 Report Writing

XIAO LINGYI (U2020057G): Task 2 Code, Task 3 Code, Task 2 Report Writing