Grab x Microsoft
Future Ready Skills Challenge 2020

Traffic Management Challenge

Done by:
Team 2 - Chuan Hao, David, Sherisse

About Us

Chuan Hao

David

Sherisse

Dr Peter Leong
(Mentor)

Overview

a. Introduction to challenge
b. Preparing for the problem statement
c. Problem Statements
d. Overall Conclusion

Introduction to challenge

Traffic Demand Dataset:

Data Schema (Field & Description):

```
- geohash6: geohash is a public domain geocoding system which encodes a geographic location into a short string of letters and digits
- day: the value indicates the sequential order and not a particular day of the month
- timestamp: start time of 15-minute intervals in the following format: <hour>:<minute>, where hour ranges from 0 to 23 and minute is
- demand: aggregated demand normalised to be in the range [0,1]
```

The dataset can be downloaded here (140MB CSV file).

Problem Statements:

The challenge targets to improve the traffic condition for road network in Southeast Asia by leveraging Grab booking demand data, which hinders mobility and economic growth. Problem statements including:

1. Which areas have high / low traffic demand?
2. How does regional traffic demand change according to day / time?
3. Forecast the travel demand for next 15min / 1hour and predict areas with high travel demand

	geohash6	day	timestamp	demand
0	qp03wc	18	20:0	0.020072
1	qp03pn	10	14:30	0.024721
2	qp09sw	9	6:15	0.102821
3	qp0991	32	5:0	0.088755
4	qp090q	15	4:0	0.074468
...	...	...	...	...
4206316	qp03y1	39	13:0	0.012731
4206317	qp097e	23	18:45	0.083179
4206318	qp03m6	32	12:15	0.123260
4206319	qp02zv	42	5:15	0.120100
4206320	qp03yv	15	4:0	0.042656

4206321 rows × 4 columns

Problem Statements

1. Which areas have high / low traffic demand?
2. How does regional traffic demand change according to day / time?
3. Forecast the traffic demand for next 15 min / 1 hour and predict areas with high traffic demand.

Preparing for the problem statement

Preparing for the problem statement

Dependencies used:

```
!pip install geolib

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

from geolib import geohash

from matplotlib.animation import FuncAnimation
from IPython.display import HTML

%matplotlib inline
```

Data Visualisation & Manipulation

Manipulating Geohash

Displaying animated visualisations

Data Preparation

- Only has 4,206,321 rows of data in the original dataset provided.
- In theory, there should be 7,782,624 rows of data.
  - 1239 unique geohash * 61 days * 24 hours * 4 quarters in one hour
- As we were instructed, we shall treat these missing rows of data as if there is no demand at all.

	geohash6	day	timestamp	demand
0	qp03wc	18	20:0	0.020072
1	qp03pn	10	14:30	0.024721
2	qp09sw	9	6:15	0.102821
3	qp0991	32	5:0	0.088755
4	qp090q	15		
...	...	...		
4206316	qp03y1	39		
4206317	qp097e	23		
4206318	qp03m6	32		
4206319	qp02zv	42		
4206320	qp03yv	15		

4206321 rows × 4 columns

	geohash6	day	timestamp	demand
0	qp03wc	1	20:0	0.004959
1	qp03wc	1	14:30	0.272885
2	qp03wc	1	6:15	0.308053
3	qp03wc	1	5:0	0.253679
4	qp03wc	1	4:0	0.267803
...	...	...	...	...
7782619	qp0d45	61	19:45	0.000000
7782620	qp0d45	61	19:0	0.000000
7782621	qp0d45	61	0:0	0.000000
7782622	qp0d45	61	17:30	0.000000
7782623	qp0d45	61	20:15	0.000000

7782624 rows × 4 columns

Filling in missing rows of data

- Only has 4,206,321 rows of data in the original dataset provided.
- In theory, there should be 7,782,624 rows of data.
  - 1239 unique geohash * 61 days * 24 hours * 4 quarters in one hour
- As we were instructed, we shall treat these missing rows of data as if there is no demand at all.

	geohash6	day	timestamp	demand
0	qp03wc	18	20:0	0.020072
1	qp03pn	10	14:30	0.024721
2	qp09sw	9	6:15	0.102821
3	qp0991	32	5:0	0.088755
4	qp090q	15		
...	...	...		
4206316	qp03y1	39		
4206317	qp097e	23		
4206318	qp03m6	32		
4206319	qp02zv	42		
4206320	qp03yv	15		

4206321 rows × 4 columns

	geohash6	day	timestamp	demand
0	qp03wc	1	20:0	0.004959
1	qp03wc	1	14:30	0.272885
2	qp03wc	1	6:15	0.308053
3	qp03wc	1	5:0	0.253679
4	qp03wc	1	4:0	0.267803
...	...	...	...	...
7782619	qp0d45	61	19:45	0.000000
7782620	qp0d45	61	19:0	0.000000
7782621	qp0d45	61	0:0	0.000000
7782622	qp0d45	61	17:30	0.000000
7782623	qp0d45	61	20:15	0.000000

7782624 rows × 4 columns

Adding a latitude and longitude based on the geohash zone

- The latitude and longitude for each geohash region refers to the rough coordinates of the centre of the region

	geohash6	day	timestamp	demand	lon	lat
0	qp03wc	1	20:0	0.004959	90.6536865234375	-5.35308837890625
1	qp03wc	1	14:30	0.272885	90.6536865234375	-5.35308837890625
2	qp03wc	1	6:15	0.308053	90.6536865234375	-5.35308837890625
3	qp03wc	1	5:0	0.253679	90.6536865234375	-5.35308837890625
4	qp03wc	1	4:0	0.267803	90.6536865234375	-5.35308837890625
...	...	...	...	...	...	...
7782619	qp0d45	61	19:45	0.000000	90.7965087890625	-5.25421142578125
7782620	qp0d45	61	19:0	0.000000	90.7965087890625	-5.25421142578125
7782621	qp0d45	61	0:0	0.000000	90.7965087890625	-5.25421142578125
7782622	qp0d45	61	17:30	0.000000	90.7965087890625	-5.25421142578125
7782623	qp0d45	61	20:15	0.000000	90.7965087890625	-5.25421142578125

7782624 rows × 6 columns

```
uniqueGeohash = train.geohash6.unique()
```

```
# Creating two empty columns that will
# hold the value of the lat and long coordinates
train_df['lon'] = ''
train_df['lat'] = ''
```

```
for geohash6 in uniqueGeohash:
  train_df['lat'] = np.where((train_df.geohash6 == geohash6), geohash.decode(geohash6)[0], train_df.lat)
  train_df['lon'] = np.where((train_df.geohash6 == geohash6), geohash.decode(geohash6)[1], train_df.lon)
```

Data preparation for time series analysis

- Splitting timestamp into hours and minutes

```
splitTime = train_df.timestamp.str.split(':', expand = True)
```

```
timeDf = train_df
timeDf['hour'] = splitTime[0].astype(int)
timeDf['minute'] = splitTime[1].astype(int)
timeDf
```

	geohash6	day	timestamp	demand	lon	lat	hour	minute
0	qp03wc	1	20:0	0.004959	90.6536865234375	-5.35308837890625	20	0
1	qp03wc	1	14:30	0.272885	90.6536865234375	-5.35308837890625	14	30
2	qp03wc	1	6:15	0.308053	90.6536865234375	-5.35308837890625	6	15
3	qp03wc	1	5:0	0.253679	90.6536865234375	-5.35308837890625	5	0
4	qp03wc	1	4:0	0.267803	90.6536865234375	-5.35308837890625	4	0
...	...	...	...	...	...	...	...	...
7782619	qp0d45	61	19:45	0.000000	90.7965087890625	-5.25421142578125	19	45
7782620	qp0d45	61	19:0	0.000000	90.7965087890625	-5.25421142578125	19	0
7782621	qp0d45	61	0:0	0.000000	90.7965087890625	-5.25421142578125	0	0
7782622	qp0d45	61	17:30	0.000000	90.7965087890625	-5.25421142578125	17	30
7782623	qp0d45	61	20:15	0.000000	90.7965087890625	-5.25421142578125	20	15

7782624 rows × 8 columns

Data preparation for time series analysis

- Splitting into weekday, weekend and weeks

```
#* A function to separate the data by their weeks
def seperateWeeks(x):
    weekVal = int(x / 7)
    if x % 7 != 0:
        weekVal += 1

    return weekVal
```

```
weekDay = []
weekEnd = []
allDays = [i for i in range(train_df['day'].min(), train_df['day'].max() + 1)]

for i in range(len(allDays)):
    currentDay = allDays[i]
    if currentDay % 7 == 0:
        weekEnd.append(allDays[i - 1])
        weekDay.remove(allDays[i - 1])

        weekEnd.append(currentDay)

    else:
        weekDay.append(currentDay)

print("weekDay: {}".format(weekDay))
print("weekEnd: {}".format(weekEnd))
```

```
weekDay: [1, 2, 3, 4, 5, 8, 9, 10, 11, 12, 15, 16, 17, 18, 19, 22, 23, 24, 25, 2
weekEnd: [6, 7, 13, 14, 20, 21, 27, 28, 34, 35, 41, 42, 48, 49, 55, 56]
```

```
#* A function to determine if the data is a weekday or a weekend
def seperateDays(x, weekDay, weekEnd):
    if x in weekDay:
        return "WeekDay"

    elif x in weekEnd:
        return "WeekEnd"
```

Problem Statement 1

Which areas have high / low traffic demand?

About the problem statement

Problem Statement 1: Which areas have high / low traffic demand?

The rough breakdown of the total region provided in the dataset when looked at in real-world terms is as follows:

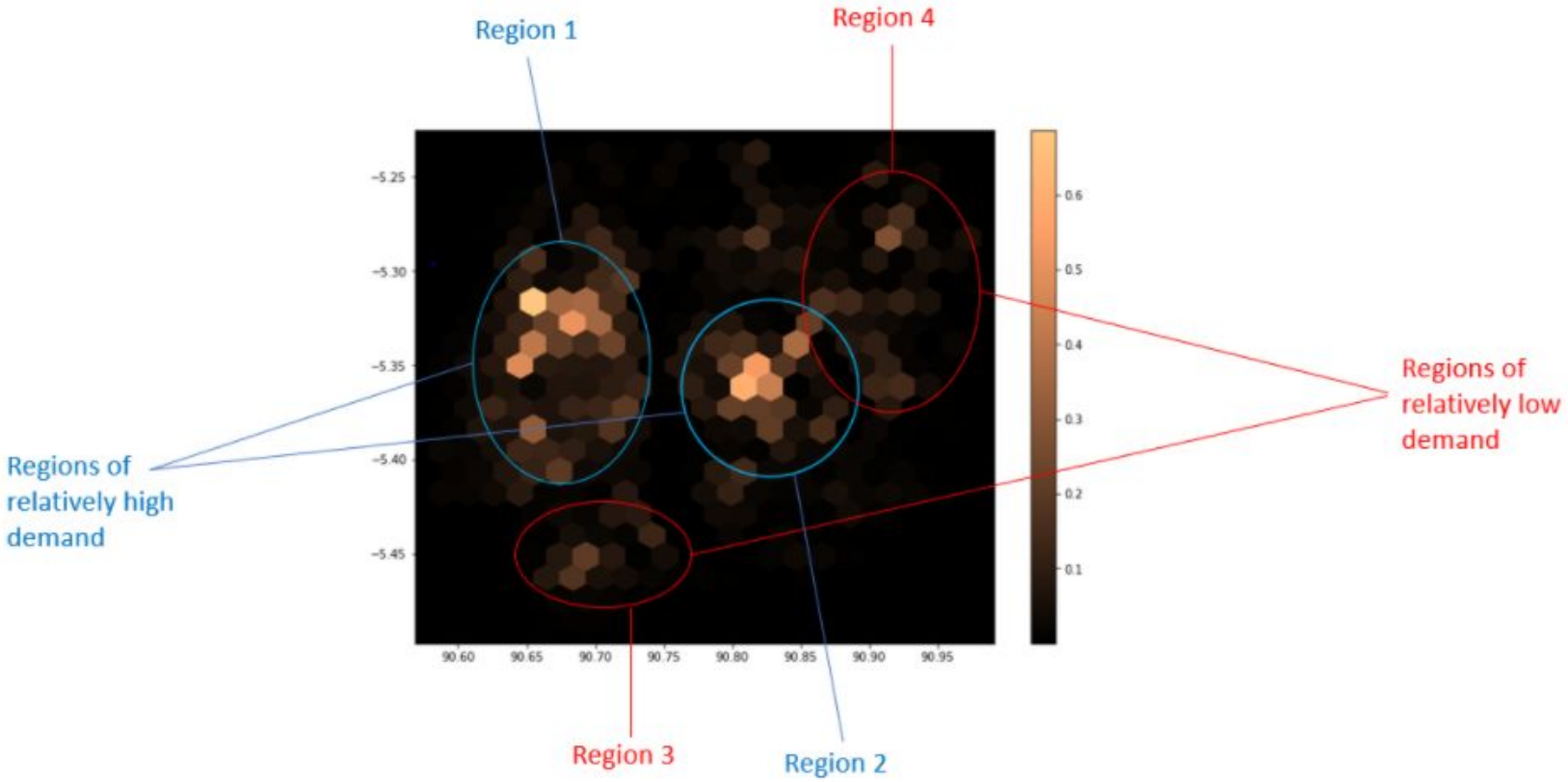
- Latitude range: 27.75 km
- Longitudinal range: 44.4 km
- Total Area: 1232.1 km square
- Area of Singapore: 725.7km square

Originally, we assumed that the dataset provided would be from Singapore. However, after looking at the range of latitude and longitude values for the total area provided in the dataset, we can see that the total area provided is larger than that of the total area of Singapore.

As such, we can probably assume that the area provided either encompasses more than a single country or multiple countries in Southeast Asia. Regardless, as we were informed that the geohash data provided would be offset, it would be more beneficial to focus on the data itself and not try to place on the World Map.

Furthermore, as the total area is quite large, we decided it would be best to first look at the demand per region broadly. As such, in order to answer this problem statement, we identified two regions each of relatively high and low demand.

Aggregated traffic demand for each region



Aggregated traffic demand for each region

Regions with obviously larger demand (Estimated lat and lon range):

```
Region1 -> ( -5.30 to -5.40 ), ( 90.60 to 90.75 )
    * 11.1km by 16.65km
    * 184.815 km square
Region2 -> ( -5.33 to -5.40 ), ( 90.75 to 90.85 )
    * 7.77km by 11.1km
    * 86.247 km square
```

Region of relatively lower demand (Estimated lat and lon range):

```
Small region below region 1 -> ( -5.40 to -5.50 ), ( 90.65 to 90.75 )
    * 11.1km by 11.1km
    * 123.21 km square
Region to the right of region 2 -> ( -5.30 to -5.40 ), ( 90.87 to 90.95 )
    * 11.1km by 8.88km
    * 98.568 km square
```

1 degree (lat and lon) == 111km

Conclusion for the problem statement

Generally, each region we identified has a smaller sub-area in which the demand is the greatest and the grids leading up to those sub-areas have a smaller demand as the distance to the sub-area increases.

Currently, we have two hypothesis.

1. Regions of relatively high demand are Central Business District (CBD) areas whereas regions of relatively low demand are Residential areas.
2. The sub-area within each region, whether of high or low demand, are CBD areas.

Problem Statement 2

How does regional traffic demand change according to day / time?

About the Problem Statement

Problem Statement 2: How does regional traffic demand change according to day / time?

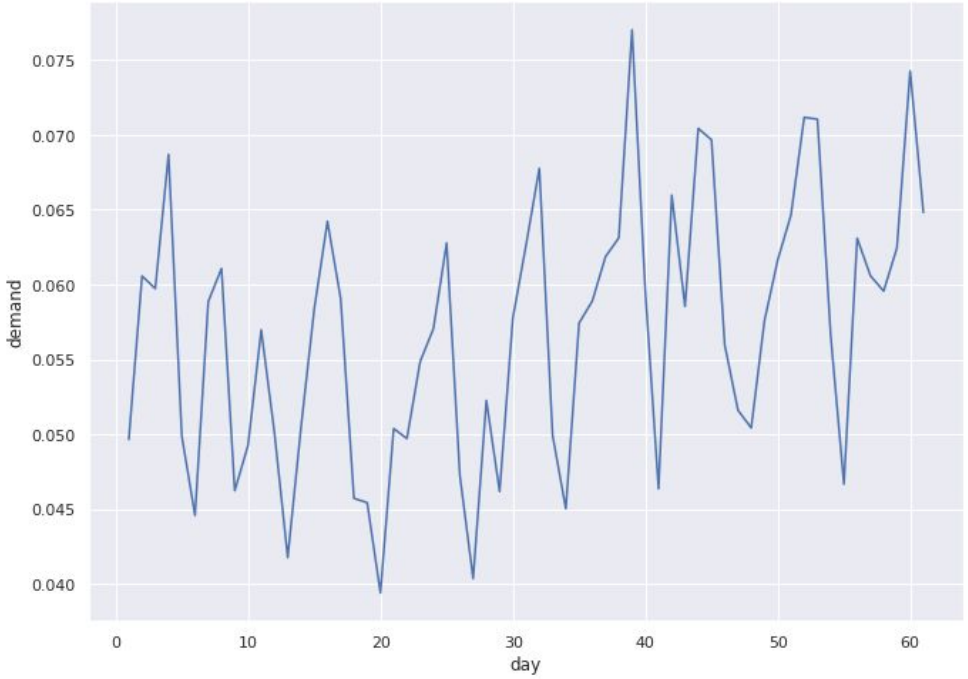
In order to answer this question, we made use of the regions of high and low demand that we identified earlier.

We also identified certain main trends that we wished to compare based on what we imagined to be true going off our experience with changes in traffic in Singapore.

Namely, we examined:

- General trends in a day
  - We theorised that the trend would follow an upside-down 'W' shape
  - Times when the demand is at its peak is during the morning and evening peak hours
- Differences in demand between weekdays and weekends
  - We theorised that the demand would be larger on weekends as compared to weekdays owing to the fact that more people would be free to travel around the city
- Differences in demand over the weeks
  - Considering that we were provided roughly 2 months and one week's worth of data, we felt that it might be possible that some of the data would coincide with the school holiday periods which would likely have higher demands

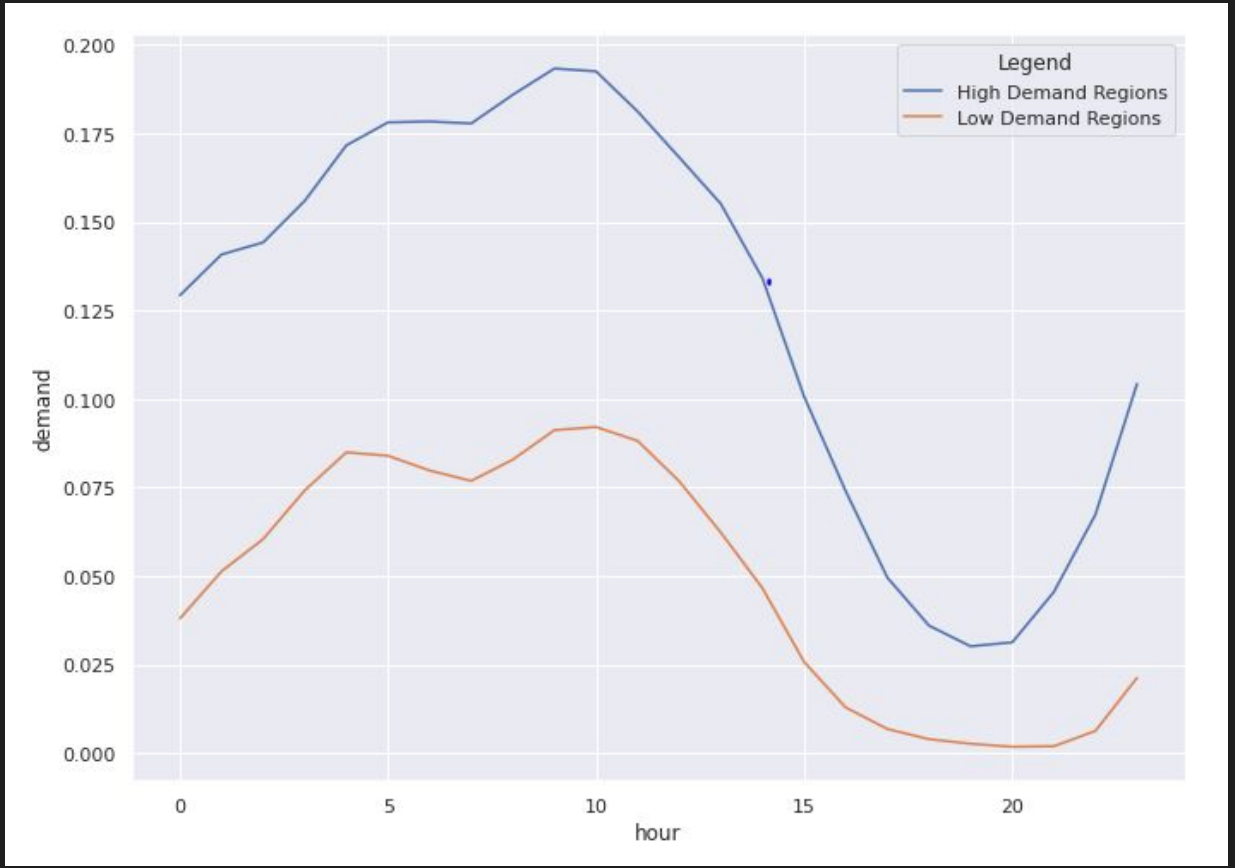
General trend throughout all 61 days



We can see a rough 'V' shape pattern in the trend over all 61 days.

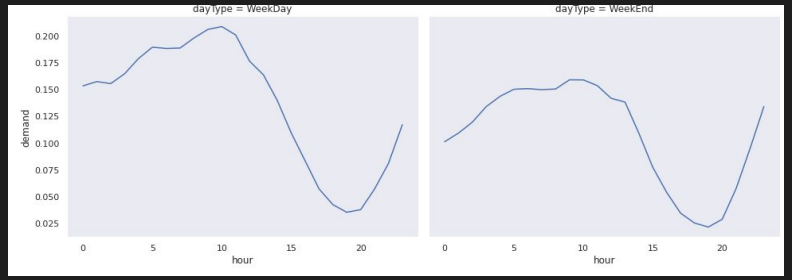
The traffic demand seems to generally decrease until around the 25th day where it starts to increase in traffic demand.

General trend in a single day

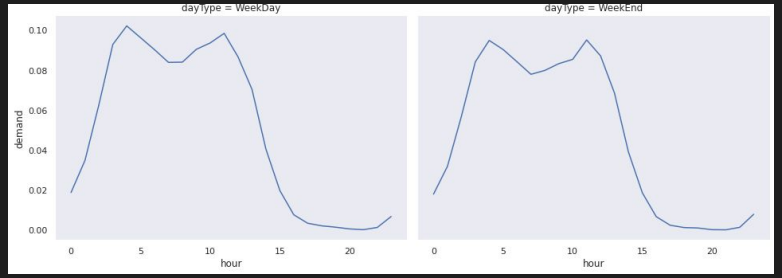


Comparing trends across all regions
Weekdays vs. Weekends

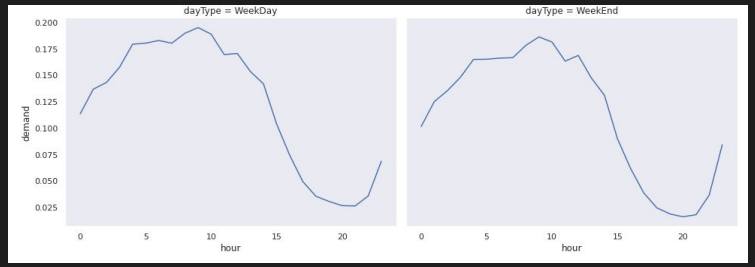
Region 1



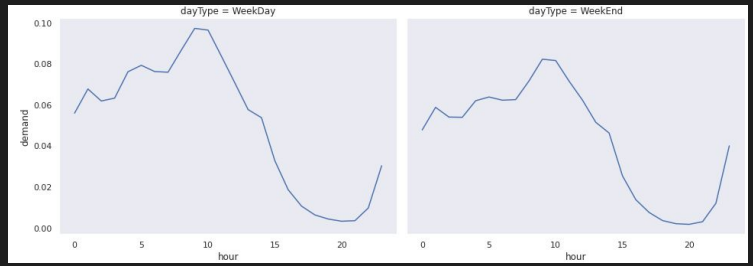
Region 3



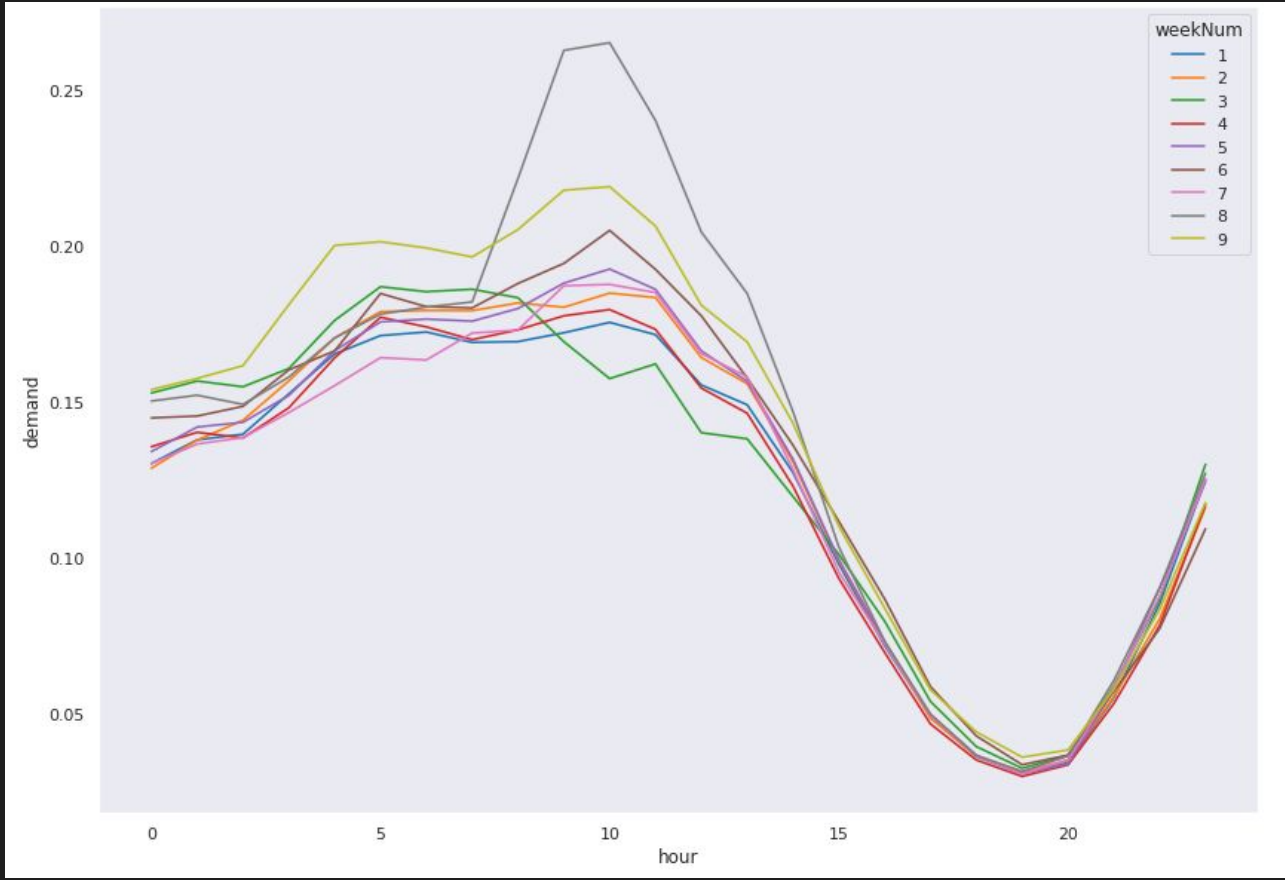
Region 2



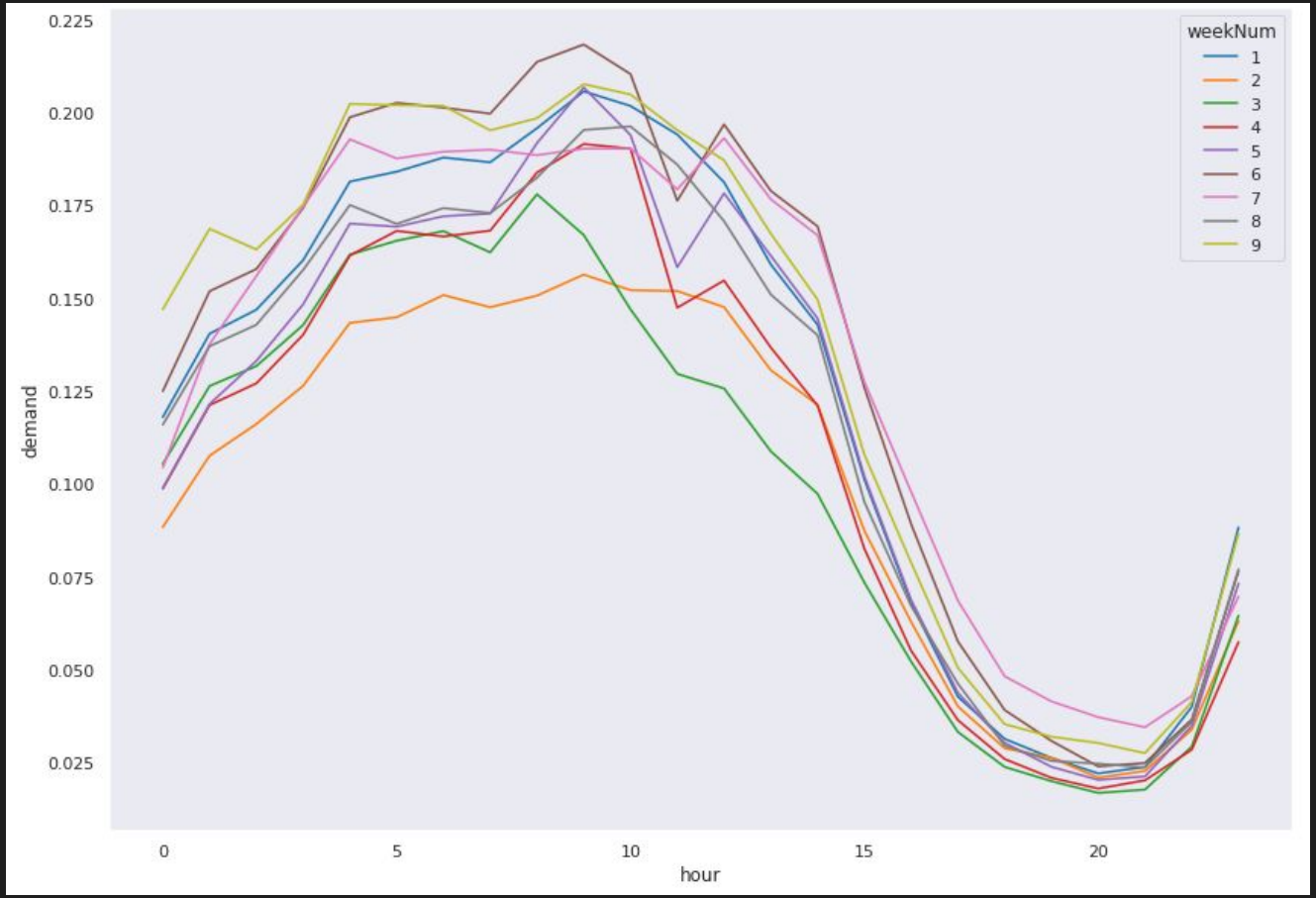
Region 4



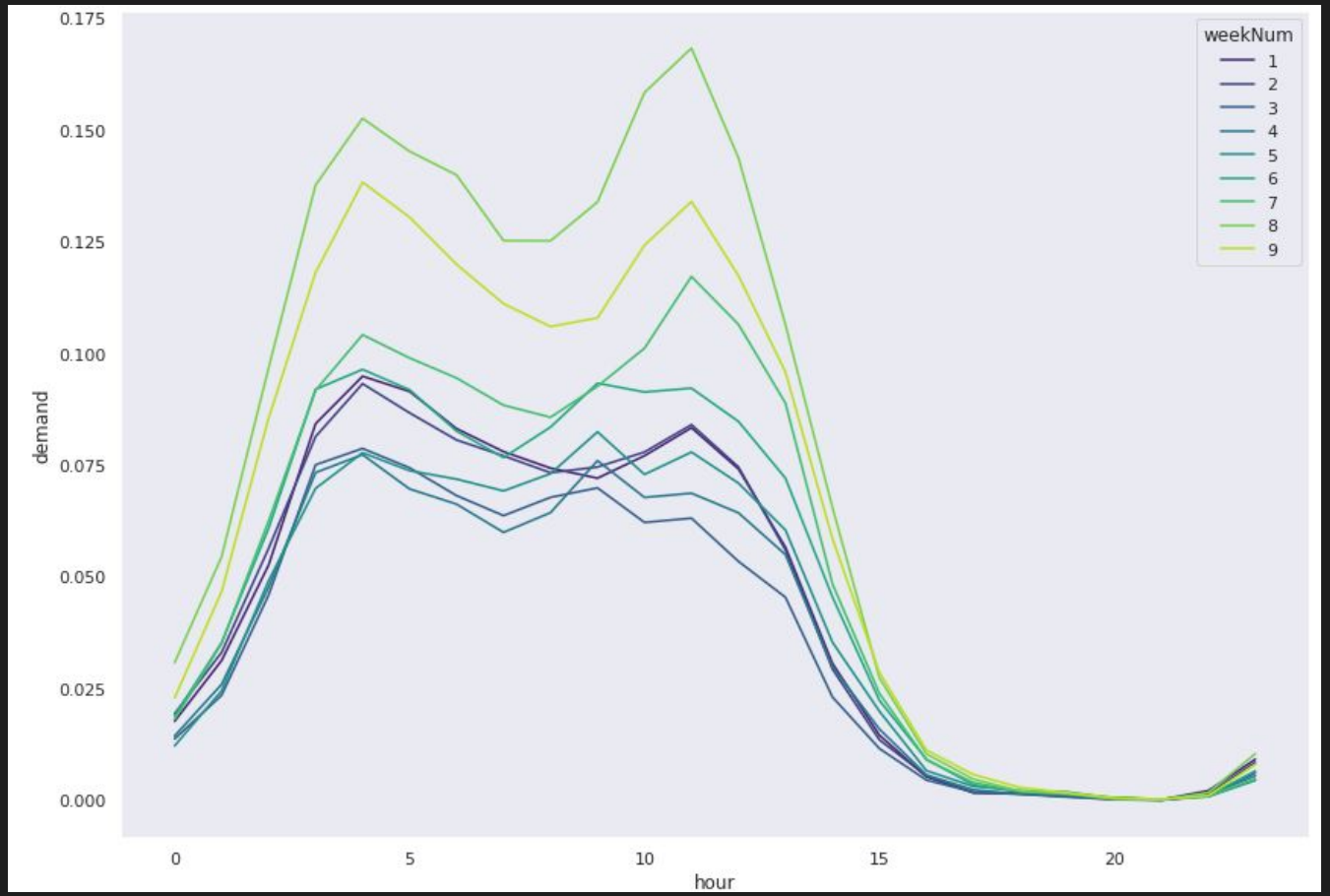
Region 1
Changes in traffic demand during different weeks



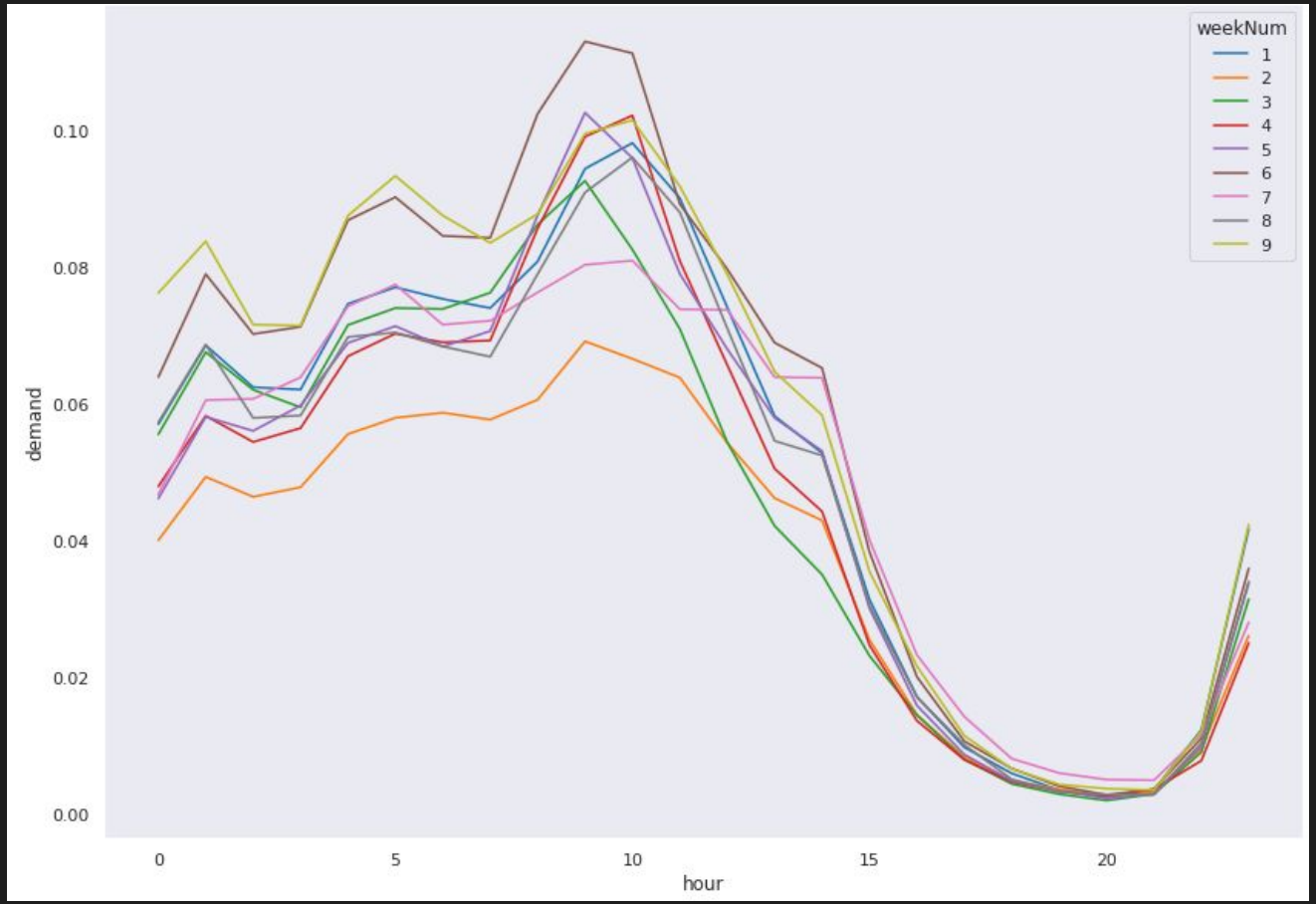
Region 2
Changes in traffic demand during different weeks



Region 3
Changes in traffic demand during different weeks



Region 4
Changes in traffic demand during different weeks



Summary

- Generally, both high and low demand regions follow the same trend.
    - Increasing demand in the morning until about 1000 hours
    - Decreasing demand from around 1000 to 1900 / 2000 hours
    - Increasing demand from 2000 until 2359
- Weekdays generally have a larger traffic demand than weekends
    - However, there is a greater spread of areas with relatively higher traffic demand during the weekends
- Most regions have fluctuations in traffic demand from week to week

Recommendations

- Similar strategies can be used regardless of whether the region has high or low demand
  - Only adjustment would be to accommodate the demand at different times
  - Adjusting prices
  - Encouraging drivers to work at these timings where there is high demand
- Weekly, probably depending on any events, could lead to different demands
  - Further investigation is required to determine if this can also be predicted

Problem Statement 3

Forecast the traffic demand for next 15 min / 1 hour and predict areas with high traffic demand.

Data Preparation (Sliding Window)

- Data from geohash of mean highest demand was used for forecasting of high demand
- Sliding window approach was used to convert the time series into supervised learning problem
- This was so that we can use supervised learning machine learning models
- Data restructured contains feature X (demand of previous timestamp) and y (demand of next timestamp)

Original Time Series

	timestamp	demand
0	00:00	0.320478
1	00:15	0.351964
2	00:30	0.386225
3	00:45	0.357434
4	01:00	0.376079
...	...	...
5851	22:45	0.188155
5852	23:00	0.167299
5853	23:15	0.181768
5854	23:30	0.230112
5855	23:45	0.212988

Restructured Data

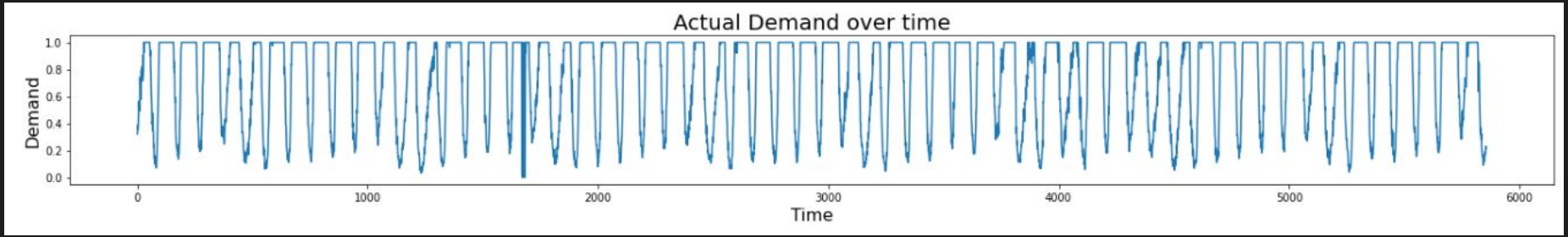
	X	y
1	0.320478	0.351964
2	0.351964	0.386225
3	0.386225	0.357434
4	0.357434	0.376079
5	0.376079	0.432797
...	...	...
5851	0.190247	0.188155
5852	0.188155	0.167299
5853	0.167299	0.181768
5854	0.181768	0.230112
5855	0.230112	0.212988

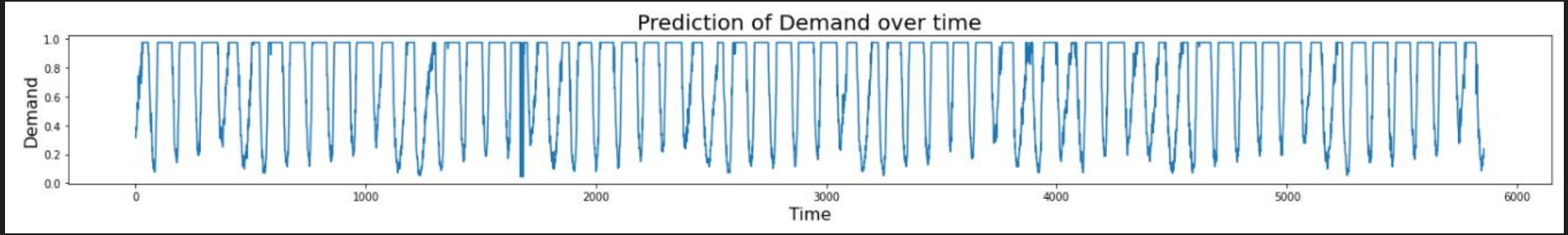
Model Training

- Restructured Data was split 80% training, 20% test
- Multiple ML regressors were used like Linear Regression, Gradient Boosting Regressor, KNN regressor, etc
- GBR has the smallest difference between the training and test scores .
- GBR would most likely be the best generaliser

	regressor	training	test	diff
2	Decision Tree	0.994205	0.958981	0.035224
3	Random Forest	0.989825	0.969631	0.020194
1	KNN	0.977169	0.974262	0.002907
6	GBR	0.975853	0.977575	-0.001721
5	Linear Regression	0.972135	0.978245	-0.006110
4	MLP	0.972053	0.978028	-0.005976
0	SVR	0.924549	0.919645	0.004904

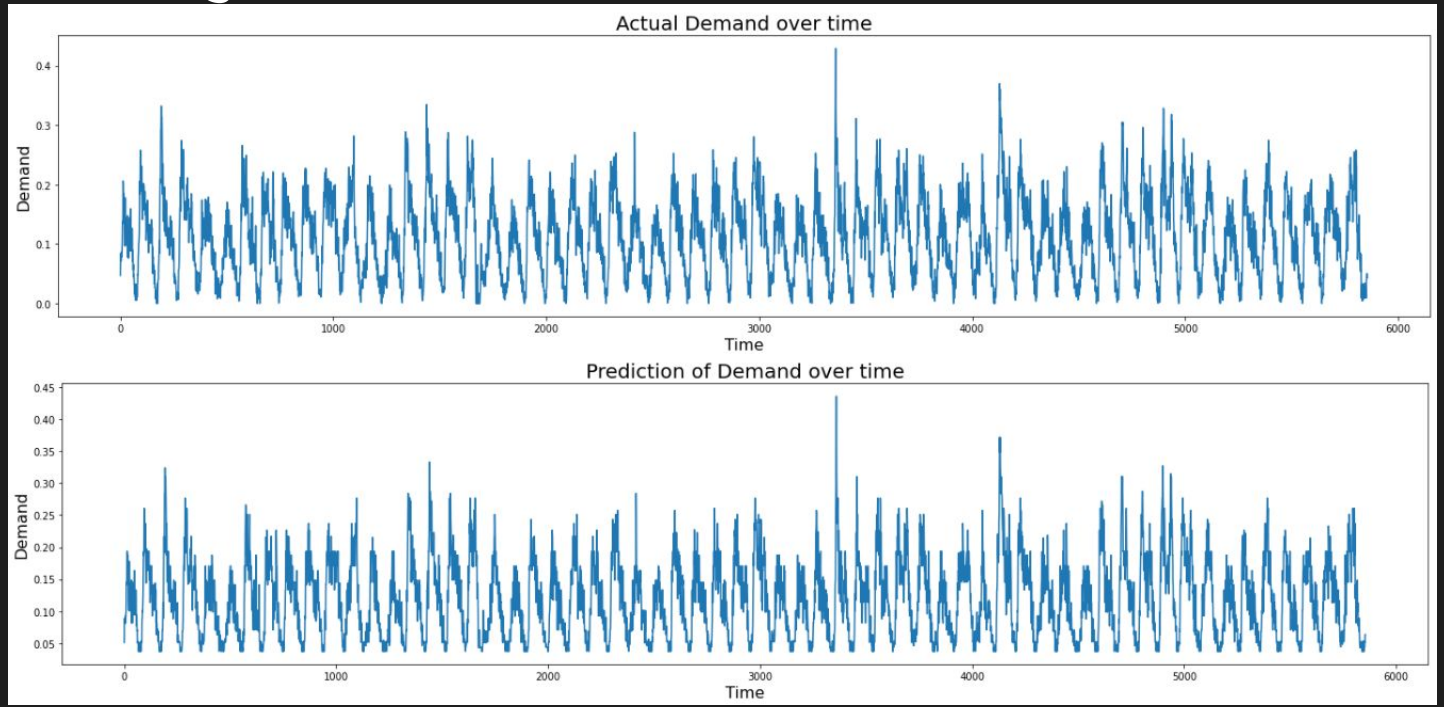
Model Training (Results)





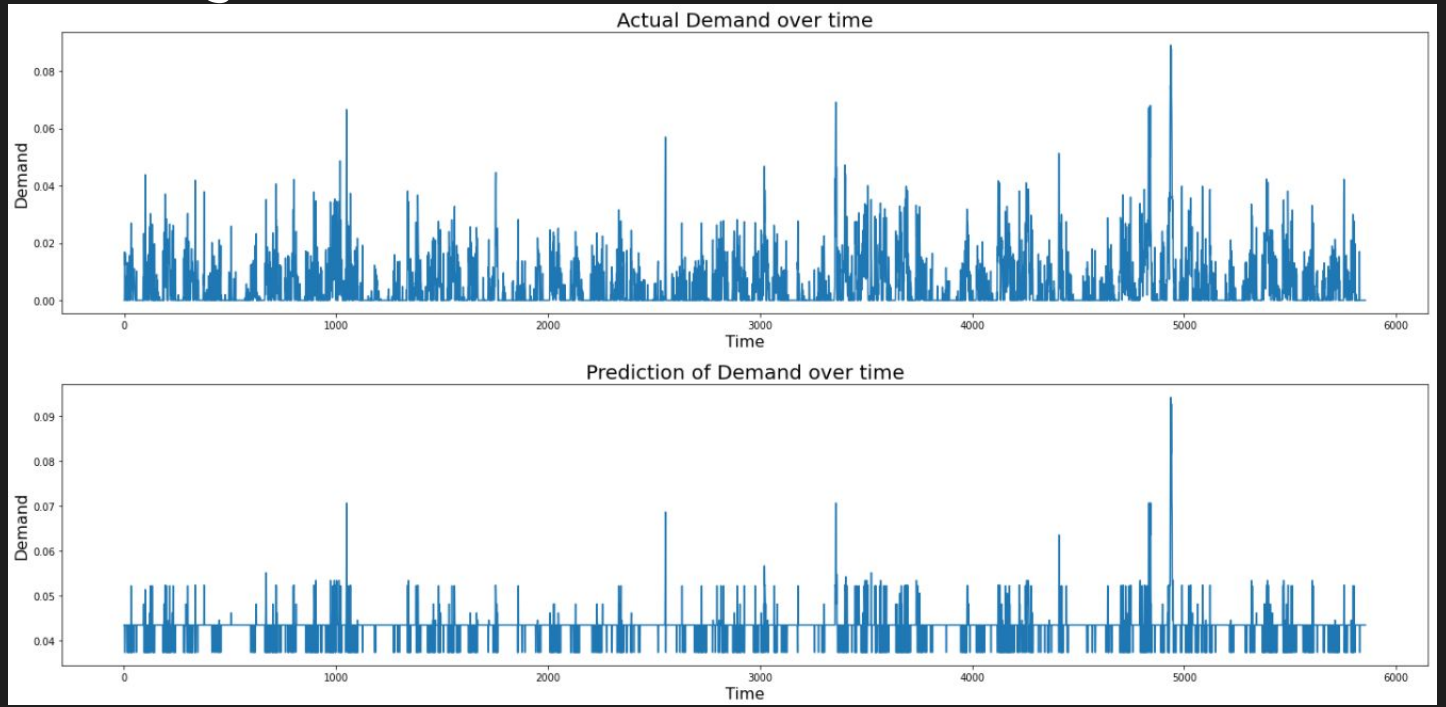
- From the graphs, the GBR seems to predict / forecast all the demand over time almost exactly like the actual demand when given the entire demand data

Model Training (Results)



- Testing the GBR out by giving it the demand of another geohash, the GBR is still able to accurately predict the demand over time

Model Training (Results)



- Testing the GBR out on a geohash relatively low a demand data, the GBR is not able to forecast geohashes for low demand that accurately, as seen from the regions significantly lower demand forecasted for the actual regions low demand.

Model Evaluation (RMSE)

```
Root Mean Square Error of GBR on training set: 0.05766612487373278
Root Mean Square Error of GBR on test set: 0.050886765165090574
```

```
count    5856.000000
mean        0.751653
std         0.327006
min         0.000000
25%         0.445603
50%         1.000000
75%         1.000000
max         1.000000
Name: demand, dtype: float64
```

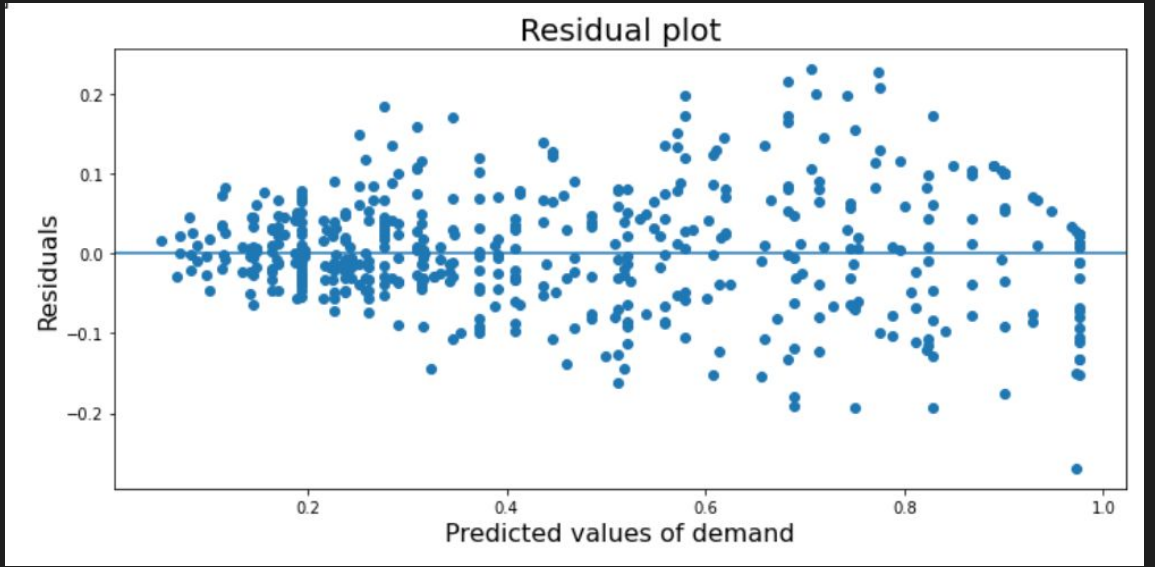
- The value of the RMSE of 0.054 on the training set and 0.057 on the test set is very small, when comparing those RMSE values to the standard deviation of the demand of the geohash, which is 0.25.
- This shows that the demands predicted do not deviate much from the actual demands. Hence this shows that the model is forecasting demand accurately with low amount of errors

Model Evaluation (Residual Plot)

From the residual plot above,

- the data points in the residual plot are symmetrically distributed, tending to cluster towards the middle of the plot.
- the data points are clustered around where y = 0, most of the residuals are within the range of -0.1 to 0.1.
- there does not seem to have any clear patterns in the plot
- the majority of the predicted values for demand by the GBR is close to the actual demand, as most of the residuals are quite close to 0

The GBR model prediction errors are quite minimal



Forecast the traffic demand for next 15 min / 1 hour and predict areas with high traffic demand.

- In order to do forecasting of the travel demand, the GBR model can be given the demand of the current time and it will output the demand of the next 15 minutes.
- The demand given (input) must be a 2-dimensional numpy array.
- The output would be a numpy array containing the predicted demand for next 15 minutes

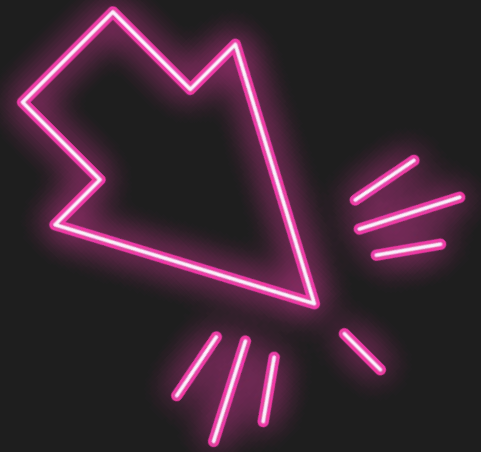
Conclusion

We managed to use the sliding approach to generate a successful model

- Furthermore, we only needed training samples from 1 geohash
- This is likely due to the fact that all regions and geohash follow the same trend. As such the model was able to learn and generalise to the trend allowing it to predict and forecast accurately.
- This was also the reason why we decided against using deep learning approaches, as we felt we did not have enough data or complexity

Overall conclusion

- We felt we did a good job exploring and solving this challenge
  - Being able to work on real data and solving real challenges was eye opening experience and challenged us to think much more than academic datasets
- There are much more details and explanations in the main notebook in the github repo.
- If you want to find out what specific code we used, you can check out the development notebooks in their respective folders as well



Thanks for listening!

Our Githubs:

@chuanhao01

@David-The-Programmer

@sherissetjw

The github repo:
https://github.com/chuanhao01/MicrosoftxGrab_FRSP2020_TM_Team_2