Assignment1, CMPT826

Step 2: Stratify and Aggregate

- Seyedeh Mina Mousavifar
- 11279515
- sem311

Obtaining saskatoon data with less than 100m accuracy and users with more than 50% battery records

In [1]:

```
import pandas as pd

# retrieving saskatoon data with less than 100m accuracy
gps_data = pd.read_pickle('data/gps_filter_final_50.pkl')

# removing unnecessary columns
gps_data = gps_data.drop(['alt', 'bearing', 'speed', 'record_time_minute', 'timestamp', 'pokemon'], 1)
```

Aggregate by time by taking the average location every duty cycle

In [2]:

```
import datetime

# sorting based on time
gps_data = gps_data.sort_values(['user_id', 'record_time']).dropna().reset_index()

# Converting record time to separate Date and Time variable
gps_data['Dates'] = pd.to_datetime(gps_data['record_time']).dt.date
gps_data['Time'] = pd.to_datetime(gps_data['record_time']).dt.time
gps_data['Hour'] = pd.to_datetime(gps_data['record_time']).dt.hour
gps_data['Minute'] = pd.to_datetime(gps_data['record_time']).dt.minute
gps_data['Second'] = pd.to_datetime(gps_data['record_time']).dt.second

# removing December test data
testdate = datetime.datetime.strptime('2016-12-09', "%Y-%m-%d").date()
gps_data = gps_data[(gps_data['Dates'] > testdate)]
```

In [3]:

```
import math

# finding study start date by finding minimum date after test date in December!
start_time = gps_data.record_time.min()
print('Study begins on', start_time)

# finding study end date by finding maximum date
end_time = gps_data.record_time.max()
print('Study ends on', end_time)

# total number of duty cycles during study
n_duty = math.ceil((((end_time - start_time).total_seconds())/60)/5)
```

```
Study begins on 2017-02-01 13:36:56.867000
Study ends on 2017-03-08 16:35:03.378000
```

In [4]:

```
# first column as each duty cycle start time
start_duty = pd.date_range(start_time, periods=n_duty, freq='5min')

# getting second item of previous dataframe as first duty cycle end time
# second column as each duty cycle end time
end_duty = pd.date_range(start_duty[1], periods=n_duty, freq='5min')

duty_num = pd.Series(range(1,n_duty+1))

duty_data = pd.DataFrame({'duty': duty_num,'start_time': start_duty,'end_time': end_duty})
```

In [5]:

```
def calc_duty(time):
    '''
    This functions find duty cycle of specific time during study
    :param time: record time
    :return: duty cycle of given record time
    '''
    result = duty_data[(duty_data['start_time'] <= time) & (duty_data['end_time'] > time)]
    if result.empty:
        print('no duty cycle')
    return result.iloc[0].duty

# finding duty cycle for gps records
gps_data['duty_num'] = gps_data.apply(lambda x: calc_duty(x.record_time), axis=1)

# saving data for future use
gps_data.to_pickle('data/gps_duty.pkl')
```

In [114]:

```
# calculating mean of latitude and longitude for every duty cycle
gps_data = gps_data.astype({'lat': 'float64', 'lon': 'float64'})
gps_data = gps_data.groupby(['user_id',
                             'duty_num']).agg(lat=('lat', 'mean'),
                                              lon=('lon', 'mean')).reset_index()
```

UTM Conversion

In [115]:

```
from pyproj import Proj

myproj = Proj('epsg:32613', proj='utm', zone=13, ellps='WGS84', preserve_units=True)

gps_data['x'], gps_data['y'] = myproj(gps_data['lon'].values, gps_data['lat'].values)

gps_data.to_pickle('data/gps_utm.pkl')
```

Grid calculation

Finding corresponding grid for each user.

In [116]:

```
import numpy as np

GRID_SIZE = 400

# find grid start point
start_x, start_y = gps_data.x.min(), gps_data.y.min()

# labeling grids
gps_data['x_grid'] = np.ceil((gps_data['x'] - start_x)/GRID_SIZE)
gps_data['y_grid'] = np.ceil((gps_data['y'] - start_y)/GRID_SIZE)
```

Heatmap plotting

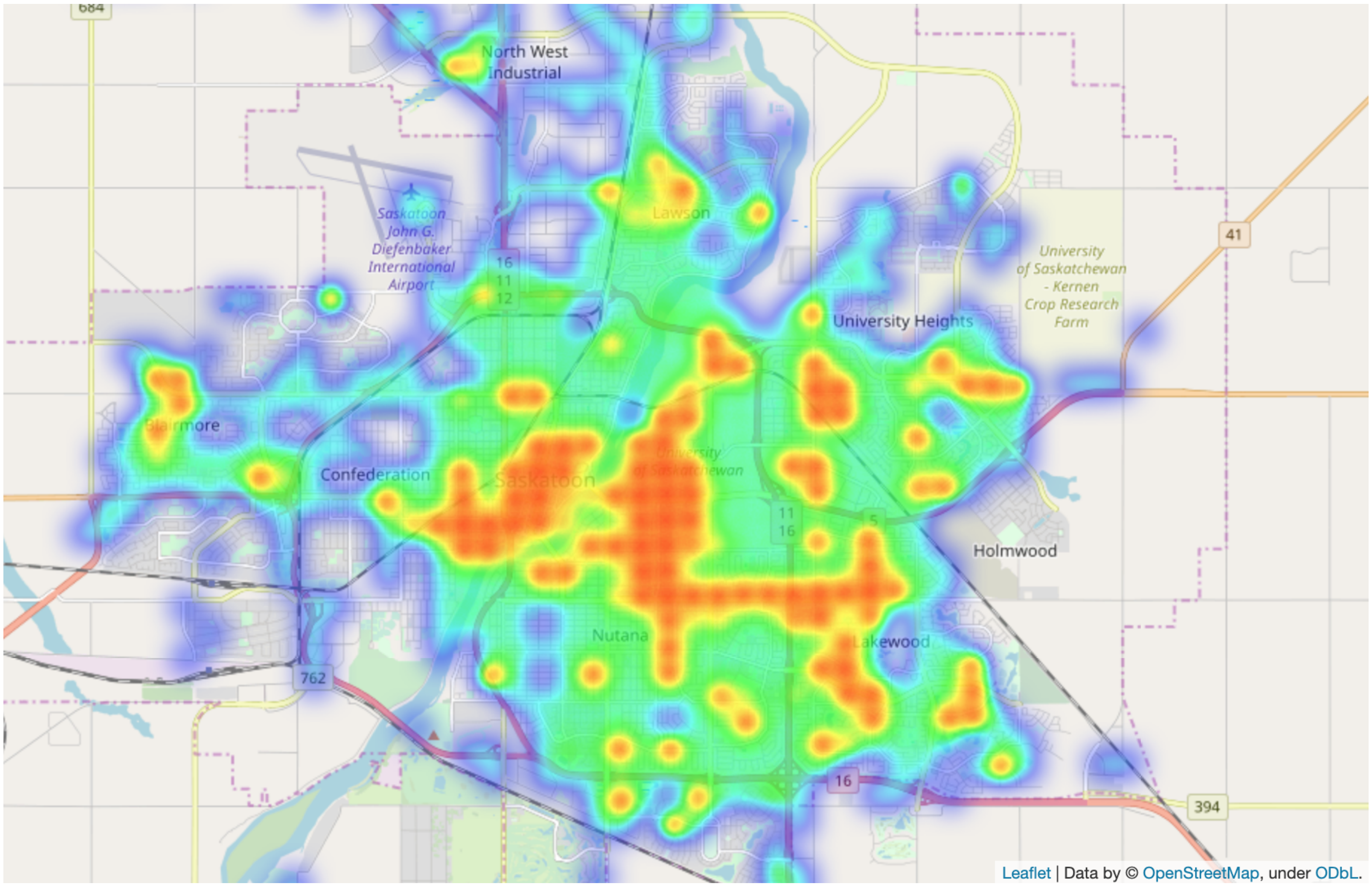
Aggregating grids for density calculation

In [117]:

```
# count number of users in each cell
gps_grid = gps_data.groupby(['x_grid', 'y_grid']).agg(grid_count=('user_
id', 'count')).reset_index()
gps_grid = gps_grid.astype({'grid_count': 'float64'})

# calculate center of grid to convert to latitude and longitude for heat
map plotting
gps_grid['x_center'] = gps_grid['x_grid']*GRID_SIZE - (0.5*GRID_SIZE) +
start_x
gps_grid['y_center'] = gps_grid['y_grid']*GRID_SIZE - (0.5*GRID_SIZE) +
start_y


# convert to latitude and longitude
myproj = Proj('epsg:32613', proj='utm', zone=13, ellps='WGS84', preserve
_units=True)

gps_grid['lon_center'], gps_grid['lat_center'] = myproj(gps_grid['x_cent
er'].values,
                                                        gps_grid['y_cent
er'].values,
                                                        inverse=True)
```

In [ ]:

```
import os
import folium
from folium.plugins import HeatMap


# creating map
hmap_data = folium.Map(location=[52.058367, -106.7649138128])

# for better plotting max grid_count is given as the max heat
max_count = gps_grid.grid_count.max()

# plotting map
hm_wide = HeatMap(list(zip(gps_grid.lat_center.values,
                           gps_grid.lon_center.values,
                           gps_grid.grid_count.values)),
                  radius=13)

# fit map zoom
hmap_data.fit_bounds([gps_grid[['lat_center', 'lon_center']].min().value
s.tolist(),
                      gps_grid[['lat_center', 'lon_center']].max().value
s.tolist()])

hmap_data.add_child(hm_wide)

# exporting map as html file
hmap_data.save(os.path.join('maps', 'sask_grid_400.html'))

hmap_data
```


North West Industrial
Saskatoon John G. Diefenbaker International Airport
Lawson
University Heights
University of Saskatchewan - Kernen Crop Research Farm
Blairmore
Confederation
Saskatoon
University of Saskatchewan
Holmwood
Nutana
Lakewood
684
41
16
11
12
5
762
394
Leaflet | Data by © OpenStreetMap, under ODbL.

What is the most commonly visited place by participants in Saskatoon?

> As we can see, University of Saskatchewan is the most visited place by participants in Saskatoon. This is rational because participants are recruited from the university.

Describe two other commonly visited places based on your heatmap.

> City center, and 8th street are the other most visited places.

Neighbourhoods

Based on the heatmap name the top three neighbourhoods where participants live. Describe the method you used to infer home locations.

> In the map, *Varsity View*, *College Park*, and *Sutherland* neighbourhood has a high number of visits, which is logical, because participants are mostly students.
>
> People spend most of their time at home(at least during the night), so these locations might have a high visit number. Based on the heatmap above, areas with more visit numbers are plotted with colours closer to red. So I scan the plot for these locations and remove industrial and administrative areas, which lead to these neighbourhoods.