CMPT 353 Project

Our project provides high-quality but cost-effective travel plans for tourists to Vancouver. Users can input the GPS coordinates of their departure and get their own recommended route depending on ratings or popularity. When on the trip, the program can get the users' position and show restaurants nearby. If the users take photos when travelling, they will know any possible places they might pass.

Files

- Data
  - osm (necessary)
- Image
  - photo (necessary)
  - pic.jpg (necessary)
- cleaning.py
- analyzing.py
- visualization.py
- main.py
- main.ipynb

Before run our project, you need have those modules installed:

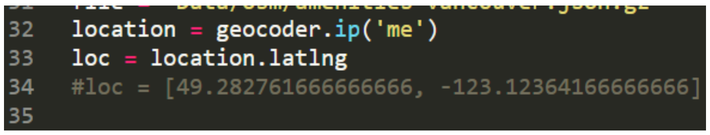
pandas
numpy
plotly
matplotlib
Pillow
nltk
wordcloud
opencage
geocoder
googlemaps
piexif
exifread
gpsPhoto
qwikidata
textblob

Important Note

When run “route” or “all”, it needs the nltk module, while an error may show that “resource stopwords not found”. Please run those commands in python3:

```
import nltk
import ssl
try:
    _create_unverified_https_context = ssl._create_unverified_context
except AttributeError:
    pass
else:
    ssl._create_default_https_context = _create_unverified_https_context
nltk.download()
```

When run “restaurant” or “all”, it uses geocoder to get your current positions, while there will be an error if you request too many times. Find those lines in cleaning.py file. You could use the fix position by replacing them with their next line to show results



```
location = geocoder.ip('me')
loc = location.latlng
#loc = [49.282761666666666, -123.12364166666666]
```

Run with command

We also provide a Jupyter Nookbook version for better visualization.

- To get recommending interesting routes:

  python3 main.py route

- To find nearest restaurants:

  python3 main.py restaurant

- To find possible amenities visited:
  (need photos with GPS information under the file Image/photo/, sample photos have been provided)

  python3 main.py photo

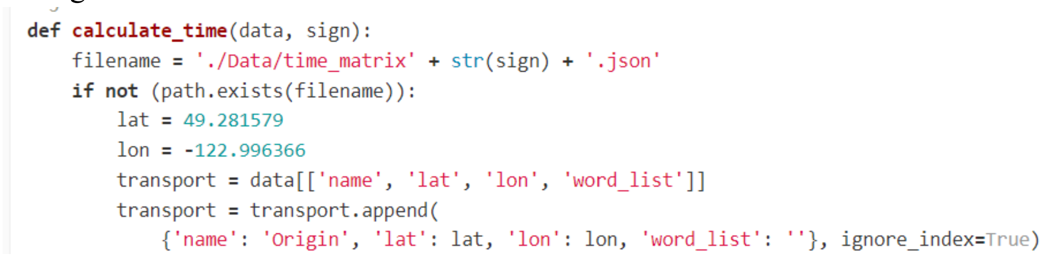
- To run all program together:

  python3 main.py all

When run command with “route” or “all”, it will output those files
addition.json, addition1.json, addition2.json,
reviews.json (containing additional information achieved from API),
data.json (used for analyzing),
route.txt (final paths and time, used for visualizing),
hot_locs.json, low.json (contain reviews word and rates for top-5 and last-5 location),
time_matrix0.json (contains transportation time for each pair of selected locations [0: means selected by rate, 1: means selected by heat]),

Customization

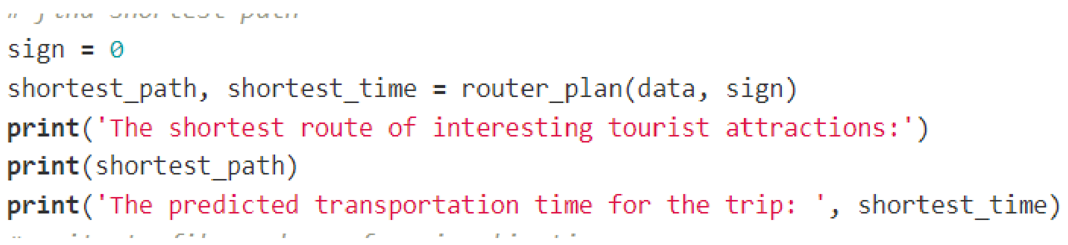
1. Change the original location for interesting route.
   Change lat and lon



```
def calculate_time(data, sign):
    filename = './Data/time_matrix' + str(sign) + '.json'
    if not (path.exists(filename)):
        lat = 49.281579
        lon = -122.996366
        transport = data[['name', 'lat', 'lon', 'word_list']]
        transport = transport.append(
            {'name': 'Origin', 'lat': lat, 'lon': lon, 'word_list': ''}, ignore_index=True)
```

2. Change the selected criteria for interesting route
   In analyzing.py,

generateRoute(output): change ‘sign’ to 1 or 0
[0: means selected by rate, 1: means selected by heat]



```
sign = 0
shortest_path, shortest_time = router_plan(data, sign)
print('The shortest route of interesting tourist attractions:')
print(shortest_path)
print('The predicted transportation time for the trip: ', shortest_time)
```