If you like to travel and experience new cities in a couple of days, you know how hard it is to choose what to see and where to go. What, if you had the all the information you need at hand with just one click?

Let's do ANY CITY in 3 days!

Capstone Project for course "IBM Data Science Professional"

Martin Koschmieder

Contents

LET'S DO ANY CITY IN 3 DAYS!

Introduction

Many people love to travel and experience new cities. These days people are always busy, so a 3 day trip is all they can carve out of their routine to get away from your everyday life and re-charge their batteries by some new experiences. Even if experiencing something new is the primary objective, many folks still like to stick to a certain routine to get the most out of their 3 days given their personal preferences.

For instance, you might want to visit the top 3 museums and sights and see different restaurants and nightlife areas. Despite the readily available information on the internet, planning a trip to fit your personal preferences can be a tedious job! There's just too much information and just sorting and selecting can keep you busy for days. That's the catch: You are too busy to take a 2 week vacation and now you realize you don't even have the time to properly prepare a fully packed 3 day trip to the city, you've always wanted to visit!!

What, if you could get a customized itinerary according to your needs for any major city of the world?

What, if this itinerary would consider user ratings of the different , so you don't step into all the tourists' traps?

What, if this itinerary could be generated with JUST ONE CLICK?

Based on location data from FourSquare.com, this project will demonstrate a tool to generate a 3 day itinerary for any major city of the world that:

1) includes the 3 top-rated museums,
2) the 3 top-rated sightseeing venues,
3) the nearest neighborhoods for breakfast, dinner and nightlife for each day
4) and the most convenient hotel for the trip
5) and requires just one click!

Data

This project uses the developers api of FourSquare.com as a source for location data.

FourSquare.com provides location data, like geographical coordinates, address information, opening hours, user ratings and more. A free subscription is available that limits daily number of calls to the database, but nevertheless is a great tool for developers of location-based apps and websites.

Methodology

The project comprised the following main steps:

1. Importing the relevant python libraries (e.g. pandas, numpy, json, sklearn, folium)
2. Retrieving the relevant venue information from FourSquare.com via the developers api (e.g. name, geographical coordinates, ratings etc.). The data was converted into dataframes for the following categories:
    a. Breakfast venues
    b. Museums
    c. Sights
    d. Restaurants
    e. Nightlife (bars and clubs)
3. Selecting the 3 top rated museums and sights, respectively.
4. Determining areas with many breakfast venues, restaurants and bars/clubs. The locations were clustered into 6 clusters for each category based on geographic coordinates. The *K-means* algorithm from the library ScikitLearn was chosen for clustering.
5. A fixed order was chosen for each itinerary to be generated:

    breakfast area → museum → sight → dinner area → nightlife area

    Based on the coordinates of the locations (museums, sights) and the areas (breakfast, dinner, nightlife) the distances between subsequent item categories of the itineraries was calculated (e.g. distance of all 6 breakfast area to all 3 museums etc.).
6. Starting with the top rated museum, itineraries were generated based on the shortest distances between the items. Euclidian distance was used as a distance measure:

$$d_{i,\,i+1} = \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2}$$

    $d_{i,\,i+1}$: distance between location i and location i+1
    x_i: latitude of location i
    x_{i+1}: latitude of location i+1
    y_i: longitude of location i
    y_{i+1}: longitude of location i

    To ensure that all itineraries were unique, the items of the first itinerary were dropped from the respective dataframes, before the next itinerary was generated etc.
7. The hotel was chosen as the best rated hotel around the most central location between the breakfast areas and nightlife areas (in order to minimize the distance

between the first and last item of the itineraries to the hotel. The location was determined as the centroid coordinates of a cluster of all breakfast areas and nightlife areas (k-mean algorithm from ScikitLearn).

8. The locations, areas and itineraries were visualized in a map using the folium library.

Details can be found in the Jupyter Notebook: https://github/...

Results

Figure 1 show some examples of the results for different cities. The maps shows the itineraries with their items numbered by their order throughout the day's itinerary (e.g. breakfast areas are no. 1, museums are no. 2 etc.).

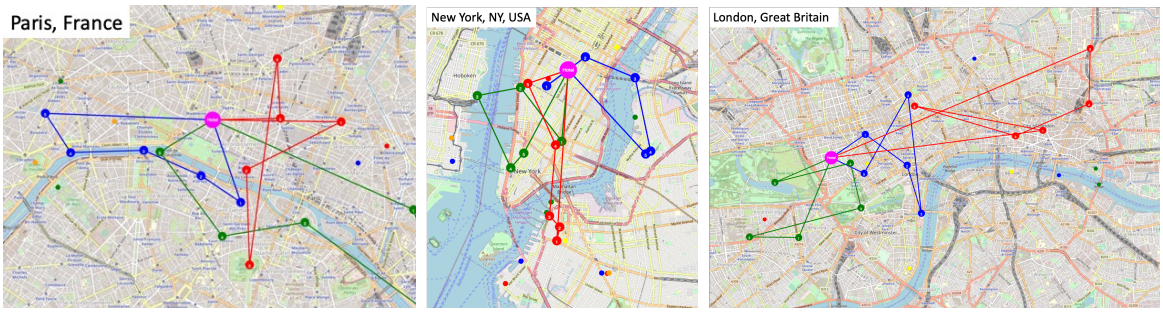

Figure 1: Itineraries for several cities

Discussion

The tool is very helpful to quickly generate itineraries for a 3 day trip to any city. The items of each day's itinerary are chosen based on a fixed daily routine and the current user ratings of each venue are considered.

However, there are some limitations to the results due to the rather simple algorithm used in this tool:

- The FourSquare.com database is used as the source of location data and user ratings. While the database is rather comprehensive for most of the major tourist cities of the world, it may lack detail for smaller or less touristy cities.

  Since only location with a user rating are considered, this could limit the locations included in the itineraries.

- The strict daily routine and the restriction that each itinerary must be unique can lead to itineraries that criss-cross across a rather larger area of the city. This is probably most evident in smaller cities with less available locations to choose from.

  Alternative algorithms could be used to avoid this, e.g. generating the itineraries based on locations from a certain areas only (instead of using the distances between items).

- The FourSquare.com developers api provides easy access to the huge location database. However, queries are sometimes rather slow to respond (e.g. retrieving venue ratings) and using a free account can be cumbersome due to the limited number of daily premium calls.

Conclusion

The project has shown that a simple tool to plan a 3 day trip to a chosen city or area can be build in Python using the FourSquare.com location data via its developers' api. Even very simple algorithms for choosing, prioritizing and combining venues into itineraries give useful results.

The itineraries can be used as is (for more adventurous travelers) or as a basis for a more detailed planning of the trip.

Improvements could be made to the tool by using more intricate algorithms, introducing more flexible routines for the itineraries or adding more features like online ordering of tickets by clicking on an item of an itinerary.

The tool could be used as is (i.e. a Jupyter notebook) or as a backbone for a mobile app or a website.