Danmarks Tekniske Universitet

DTU

Computational Tools For Data Science 02807

Authors

[redacted]

Link to the Github repository

November 28, 2023

Contents

1 Introduction **1**
1.1 Motivation and explanation 1
1.2 Approach and intro to methodologies 1

2 Dataset **1**
2.1 NomadList dataset 1
2.2 Flickr scraped dataset 1

3 Methodologies **2**
3.1 Item based recommendations 2
3.1.1 Similar items 2
3.1.2 Clustering 3
3.2 User based recommendations 3

4 Conclusion **4**

References **6**

Contributions

Table 1: List of contributions

Task	Responsible	Controlled by
Project idea	[redacted]	
0 Abstract		
1 Introduction		
2 Nomad List dataset		
2 Merge with SDG dataset		
s2 Flickr dataset		
2 NomadList EDA		
2 Flickr EDA		
3 Similar Items		
3 Clustering		
3 Plot comparing query row and prediction		
3 World map with clusters		
3 User based recommendations		
4 Conclusions		
5 Appendix		

Abstract

Nowadays digital nomadism is a widespread phenomena, affecting around 30 million people. Picking the best possible city to live in is not an easy task, so in our project we want to help this category of people by creating a recommendation system for cities. In doing so, we analyze data from Nomadlist and Flickr with techniques such as similar basket analysis, clustering, and collaborative filtering. Our results yield good recommendations in item-based recommendations. On the user-based side, the performance is hard to evaluate objectively. Compared to a state of the art model, we can say that the recommendations do not agree often.

The comparison of similarity measurement techniques, including Jaccard similarity, Cosine Similarity, and Minhashing, revealed that Minhashing exhibited superior computational efficiency with fast execution times and minimal memory usage.

1 Introduction

1.1 Motivation and explanation

With the increase in travel possibilities due to the globalization, being able to see and live in other countries and getting in touch with various cultures has become easier and easier. This phenomena scaled up to a point where, especially in the last couple of years thanks to Covid-19, people have started working from a different country [1]. The people belonging to this category are called **digital nomads**. It's not always easy for digital nomads to find the best place to move to without gathering data about the desired location, such as cost of living, safety of the city and many more. Our project aims to develop a recommendation system designed to guide digital nomads in selecting their ideal living destinations based on their unique preferences and needs.

1.2 Approach and intro to methodologies

In order to perform the above mentioned recommendations, we will make use of different techniques. First, we'll make recommendations computing the **most similar items** in our dataset, finding the cities that have similar features. Then, we will use a different technique, such as **clustering**, and compare the two results about similar cities. Given a fictional user profile, using the results provided by these two methods, we will be able to provide meaningful recommendations to the user. These two techniques will fall under the name of **item based** recommendations from now onward. Similarity of the characteristics, though, is not enough in order to provide suggestions to the user for his next destination. Very often cities from the same country have similar characteristics, but this doesn't imply that someone would only travel to the country he likes the most. Our assumption is that people enjoy **exploring**, seeing different cultures and countries. Because of that, we have to go beyond the features a city has and try to make a recommendations with other **people with similar likings** travel-wise. To perform this, we will be using a technique called **collaborative filtering**. We will call this procedure under the name of **user based** recommendation.

2 Dataset

2.1 NomadList dataset

The main dataset we are using to perform the aforementioned tasks is the NomadList dataset. It comprises information about 730 cities around the world provided by digital nomads and remote workers. This dataset includes essential features such as cost of living, quality of life, work and connectivity, amenities and entertainment, air and environmental quality, as well as cultural and social factors.

2.2 Flickr scraped dataset

In order to perform user based recommendations, we needed to gather data about users. This turned out to be a particularly challenging task, since nowadays most of user-related data is not free anymore. We decided to gather data from Flickr, a famous website used mostly to upload pictures. Using an API key, we downloaded data about pictures with title or tag matching the string *nomad*. For coherence purposes, we only got data about pictures taken in a radius of 30km from the cities in the nomadlist dataset. In order to do this, we needed to extract city names from the *place_slug* field of the nomadlist dataset, and get their coordinates using the *geopy* package.

After the scraping, we ended up having 8937 users and 18237 city visits (2 per user), each one with one or more photos taken by the user.

3 Methodologies

3.1 Item based recommendations

3.1.1 Similar items

The first step we made was the standarization of the data; this method is crucial in data science, ensuring that all variables contribute equitably and eliminating biases caused by different units or scales. In Jaccard, Cosine similarity and Minhashing, the Nomad dataset together with the SDG columns shall be used.
In this part of the report we will try to develop a recommender system based on similarity measures such as: cosine and Jaccard [2].

Jaccard similarity

For this part, some of the variables in the dataset were categorised, converting them into categories showing, for example, that a city could be 'Very safe' or 'Not safe'. In the experiment, the calculation of Jaccard similarity took around 390 milliseconds with a memory usage of approximately 1.98 MB. The query row belongs to a European city (Kaunas, Lithuania) and the recommender system in this case recommends Bruges, Belgium. Therefore the similarity between the sets is high. The system seems to recommend correctly.

Minhashing

The cost in time to calculate this experiment is 46.87 ms and the cost in memory is 4.79 MB. This evidence why Minhashing is computationally efficient because it drastically reduces the dimensionality of sets. As can be seen in the appendix Figure 1, the Minhashing comparison figure shows the similarities found between the query and the prediction. Furthermore, the recommendation is correct because the query presented the city Penang and the system has recommended the user Kuching, both belonging to Malaysia. Therefore the minhashing technique shows good results.

Cosine Similarity

In this experiment the user-input values for "costOfLiving," "fun," and "safety" underwent imputation of NaN values, feature scaling, and cosine similarity calculation. These operations took approximately 1.99 ms, 4.27 ms, and 3.26 ms, respectively, with corresponding memory usage of 0.0117 MB, 2.09 MB, and 4.09 MB. Based on user preferences, entered by command line, for example, for an input of costOfLiving=1, fun=2, safety=3, the recommended city is Berlin, Germany. This makes sense, as can be seen in the data set, the city is usually expensive, it is a city with somewhat fun and it is very safe.

Evaluation. Regarding the **computational efficiency** in this section, Minhashing demonstrated the best computational efficiency, with a relatively fast execution time and minimal memory usage. It should also be noted that the SDG columns are relevant when recommending cities in our

recommendation system. Tests were done without these columns and the best results were obtained when they were present in the dataset.

3.1.2 Clustering

Various clustering techniques, including K-Means, Agglomerative, DBSCAN, and Spectral Clustering, are employed and in terms of assessing the outcomes, we evaluate the performance, apply metrics like the Davies-Bouldin Index and the Silhouette Score but also anticipate that the clusters should align with common sense. For example, cities from the same country are expected to be grouped together. Thus, we aim to identify a suitable number of clusters that effectively organize the data into cohesive and evenly distributed groups, enhancing the precision of the recommendations.

Evaluation. DBSCAN struggled with the density variations, and Spectral Clustering had long execution times, making both unsuitable for our dataset, leading to their exclusion from the clustering task.

Regarding K-Means and Agglomerative Clustering, the latter performed better for this data set, although it should be expected to be slower when using bigger data-sets in comparison to the former. Both algorithms yielded similar results when aiming for two clusters which was the optimal k according to the metrics, classifying cities into categories that resemble "Western/Developed" and "Non-western/Developing" countries as illustrated in Figure 2.

Initially we aimed for more granular data, but we found that the metrics were not in favour for either K-Means - although the clusters are evenly distributed - nor for the Agglomerative algorithm - the result was not evenly distributed - so in that regard, none of them performed specially well due to the nature of the data. It is worth mentioning that in the granular attempt, K-Means' clusters represented countries that are often grouped together, for instance, the Nordic countries, Canada and Australia had most of their cities in the same cluster, which is often seen in various rankings.

In conclusion, the optimal use of clustering considering our scenario would be for unsupervised learning aiming to categorize the data and use it as an extra dimension.

3.2 User based recommendations

For user based recommendations, as mentioned, we used data scraped from Flickr. As described in [3], we used collaborative filtering. Collaborative filtering is a method of making automatic predictions (filtering) about the interests of a user by collecting preferences or taste information from many users (collaborating). This problem is usually solved using **matrix factorization**. In our problem, collaborative filtering would require the construction of a matrix R= N $\times$ M, where N is the number of users, and M the number of cities. Matrix R is really sparse, since most people in our dataset have lived in 2 to 5 cities. Matrix factorization allows us to fill those empty elements in the matrix with numbers corresponding to the prediction of how much the user will like a specific city. Furthermore, it allows to represent this matrix, which is really big, using two smaller matrices M and N. The dimension of these two matrices depends on a predefined latent space K. The matrices are initialized randomly, and then tuned iteratively using stochastic gradient descent with regularization. Stochastic gradient descent uses mean squared error as adjustment value. We factorized 2 different matrices:

- One with boolean values 0/1 telling us if a user had lived in a city

- One with float values representing how much a user liked a city based on the amount of photos he took in such city (normalized by that users' average number of taken photos per city)

Evaluation was made comparing to a state of the art model from *LightFM* library, in order to highlight the importance of resource-saving techniques. Resource-wise, the latter took 0.015s per loop, while our algorithm took 4s per loop. The matrix representation for SoTA model took 18237*3*32 = 1750752 bits = 0.218844MB (tuple sparse matrix representation), whilst our occupied 730*8937*32= 208768320 bits = 26.09604MB (full matrix representation). The recommendations of the two algorithms were compared to see common recommendations. As it turns out, there is very little overlap. This probably happened because:

- The dataset is not the most suited to perform this task. We thought that photos locations could be a good reference to see where nomads live, but it might not be. Also, the rating based on the amount of photos a user took in a place might not be an accurate metric.
- The training loop takes too much time to be run, hence resulting in a bad matrix factorization model.

4 Conclusion

Building a novel and original recommendation system is a not trivial task. Many were the challenges we encountered starting with gathering a sufficient amount of interesting data. This made it necessary to combine different datasets and address potential inconsistencies. We experiment with two types of recommendation systems: item-based and user-based. Overall the approach based on similar items was the most successful in terms of giving coherent and 'useful' advice. However, this approach would probably not be optimal in a real-case scenario but still can be an option in the case of limited computational resources. Regarding clustering, our aim was for finer-grained data, but we observed that neither the K-Means algorithm, despite its evenly distributed clusters, nor the Agglomerative algorithm, which produced unevenly distributed results, yielded favorable metrics. Therefore, none of the algorithms performed well in this context, reflecting the challenging nature of the data. Therefore we believe that the optimal use of clustering considering our scenario would be for unsupervised learning. The goal shifts from achieving granular distinctions to categorizing the data based on inherent patterns. By treating the clustered data as an extra dimension, we can obtain a more detailed and refined understanding of them. On the other hand, the user-based recommendation model yielded less satisfactory results. The discrepancy with the SoTA model stems from both the scarcity of input data and resources. The state-of-the-art algorithm is faster and takes up much less space than our model. This makes us reflect on the impact of certain optimization choices - in this case the use of a tuple sparse matrix representation rather than a full matrix representation - which can make a difference of 100 times, in the context of a relatively small dataset as ours. However, considering a approach user-based gave us the opportunity to implement a collaborative filtering algorithm from scratch. Overall the techniques learned throughout the course, and that we have used in this project, proved to be effective in reducing the time and resource consumption in all our experiments.

Appendix

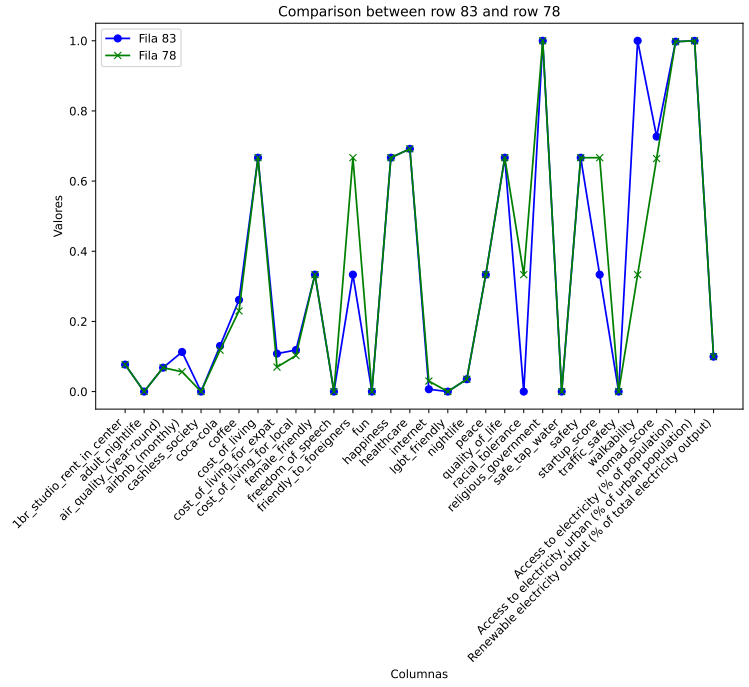

Figure 1: Comparison between query and prediction.

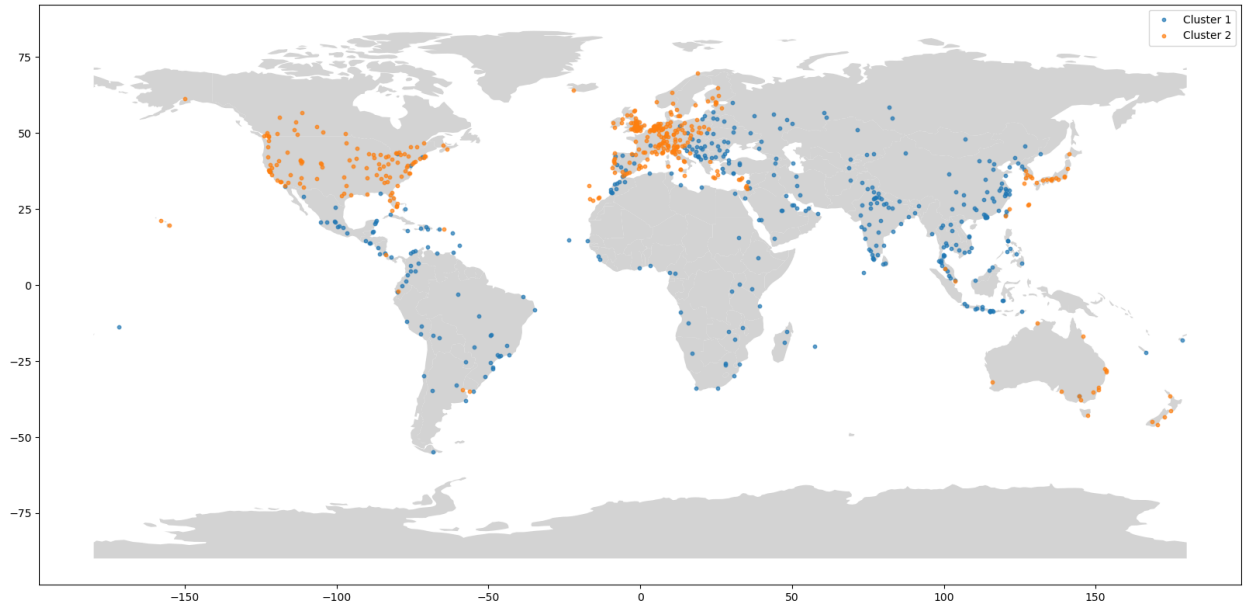

Figure 2: Clustering results for $k = 2$ in K-Means

References

[1] G. Richards, "Growth and developments in the digital nomad market since covid-19," 01 2023.

[2] V. Thada and V. Jaglan, "Comparison of jaccard, dice, cosine similarity coefficient to find best fitness value for web retrieved documents using genetic algorithm," *International Journal of Innovations in Engineering and Technology*, vol. 2, no. 4, pp. 202–205, 2013.

[3] A. Rajaraman and J. D. Ullman, *Mining of massive datasets*. Cambridge University Press, 2011.