Analysis of Vodafone users' fluxes

Scientific Computing with Python, Final Project

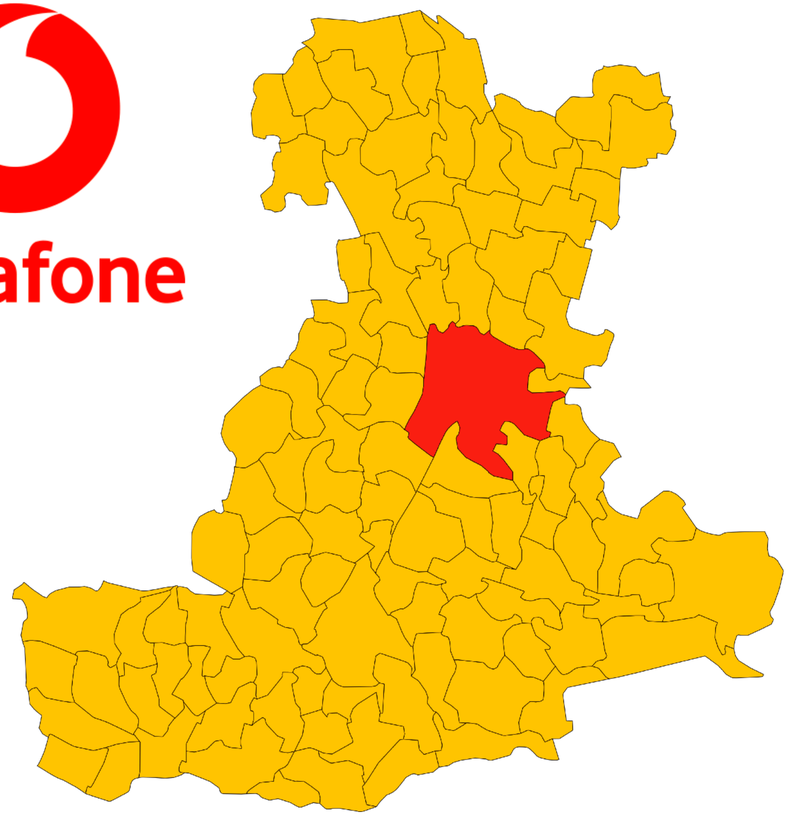
14.09.2023 | DARIA NIKOLAEVA

Introduction

The project focuses on analysis of Vodafone users' fluxes in Padova from February to March 2018.

Analyzing the movement of individuals within urban areas holds significant importance in gaining a comprehensive grasp of emerging challenges in local transportation and exploring opportunities for enhancing infrastructure and public transit systems.

vodafone



python™

jupyter

1. Data preparation

For analysis the dataset **distinct_users_day.csv** was used.
The data is provided with details of the month, type of user (resident in Padova/Italian visitor/ foreign visitor), country of provenance, together with the province and comune of the user (if available).

In addition:
codici_istat_provincia.csv: lookup file containing the mapping between province ISTAT code-names
codici_nazioni.csv: lookup file containing mapping the country code to its name
DCIS_POPRES1.csv: dataset consisting of data about Italian population per province. Downloaded from *https://www.istat.it/en/analysis-and-products/databases*

	DOW	CUST_CLASS	COD_COUNTRY	COD_PRO	PRO_COM	VISITORS
0	Mercoledì	visitor	222.0	35.0	35033.0	968
1	Lunedì	visitor	222.0	22.0	22098.0	64
2	Domenica	visitor	222.0	52.0	52032.0	516
3	Giovedì	visitor	222.0	108.0	108009.0	128
4	Giovedì	visitor	222.0	29.0	29048.0	512
...	...	...	...	...	...	...
12840	Mercoledì	foreigner	259.0	NaN	NaN	176
12841	Sabato	foreigner	602.0	NaN	NaN	164
12842	Giovedì	foreigner	732.0	NaN	NaN	80
12843	Giovedì	foreigner	297.0	NaN	NaN	80
12844	Domenica	foreigner	748.0	NaN	NaN	60

12845 rows × 6 columns

1. Data preparation

To ensure all the datasets were decoded correctly, the chardet library was used.
This library allows to analyze the source file and detect the encoding algorithm for each.

```
#Using chardet library create a function to define encoding algorithm for each csv file.

def open_dataset(file_name):

    with open(file_name, 'rb') as f:
        result = chardet.detect(f.read())

    df = pd.read_csv(file_name, encoding=result['encoding'], sep=',')
    return df
```

```
#Creating dataframes from each file to make the work with dataset easier.

day_od_df = open_dataset('day_od.csv')
distinct_users_day_df = open_dataset('distinct_users_day.csv')
codici_istat_comune_df = open_dataset('codici_istat_comune.csv')
codici_istat_provincia_df = open_dataset('codici_istat_provincia.csv')
codici_nazioni_df = open_dataset('codici_nazioni.csv')

population_df = open_dataset('DCIS_POPRES1_23082023181232317.csv')
```

1.1 Ranking of visitors from foreign countries

1. Extract Italian Country Code:

```
#extract the italian code from the national codes dataframe
italian_code = codici_nazioni_df.query(f'COUNTRY_NAME_IT == "Italia"')['COD_COUNTRY'].values[0]
```

2. Filter Foreign Visitors Dataframe:

```
foreign_visitors_df = distinct_users_day_df[distinct_users_day_df['COD_COUNTRY']
                                            != italian_code].drop(columns=['COD_PRO', 'PRO_COM'])
#filter original dataframe by country code (delete italian code) and drop useless columns, create a new dataframe
foreign_visitors_df
```

3. Create Top 20 Foreign Countries Dataframe:

```
#obtain the dataframe with top 20 in number of visitors countries
top_foreign_countries_df = foreign_visitors_df.groupby('COD_COUNTRY').sum().sort_values(by='VISITORS',
                                                                                        ascending=False).head(20)

#merge the previous dataframe with country codes to easier graph creation
top_foreign_countries_df = top_foreign_countries_df.merge(codici_nazioni_df,
                                                          left_on='COD_COUNTRY', right_on='COD_COUNTRY')

#show final dataframe
top_foreign_countries_df
```

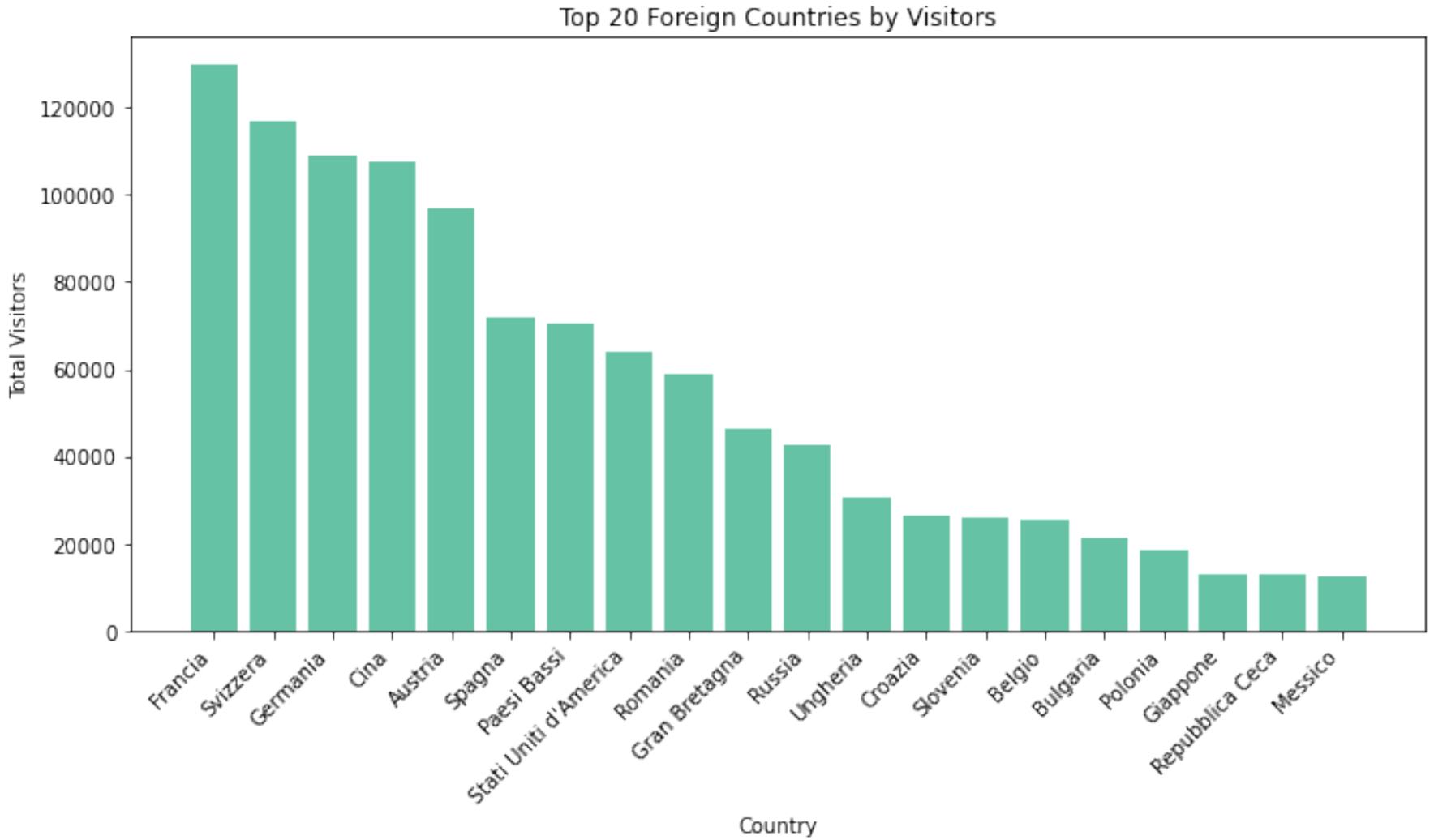
1.1 Ranking of visitors from foreign countries

4. Create a plot from data frame

```
#create a ranked plot of the first 20 countries with the most visitors

plt.figure(figsize=(10, 6))
plt.bar(top_foreign_countries_df['COUNTRY_NAME_IT'], top_foreign_countries_df['VISITORS'], color='#66c2a5')
plt.xlabel('Country')
plt.ylabel('Total Visitors')
plt.title('Top 20 Foreign Countries by Visitors')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
```

1.1 Ranking of visitors from foreign countries

5. Analyze the results:



1.2 Ranking of Italian visitors by province, weighted by the number of inhabitants

1. Filter dataset consisting of the amount of Italian population per province:

```
#data preparation: filter data and preserve only interested columns

filtered_df_1 = population_df.loc[population_df['ETA1'] == 'TOTAL']
filtered_df_2 = filtered_df_1.loc[population_df['Gender'] == 'total']
final_population_df = filtered_df_2[['Territory','Value']]

final_population_df = final_population_df.rename(columns={'Territory': 'PROVINCIA', 'Value': 'POPULATION'})

final_population_df
```

2. Filter dataset to extract only Italian visitors and merge it with the name of province:

```
#filter data and preserve only visitors

italian_visitors_df = distinct_users_day_df[distinct_users_day_df['CUST_CLASS']
                                    == 'visitor'].drop(columns=['COD_COUNTRY', 'PRO_COM'])
```

```
#calculate number of visitors per province

top_italian_visitors_df = italian_visitors_df.groupby('COD_PRO').sum()
```

```
#merge with the province codes

top_italian_visitors_df = top_italian_visitors_df.merge(codici_istat_provincia_df,
                                                        left_on='COD_PRO', right_on='COD_PRO')
```

1.2 Ranking of Italian visitors by province, weighted by the number of inhabitants

3. Create a dataframe with TOP 20 provinces weighted by amount of population:

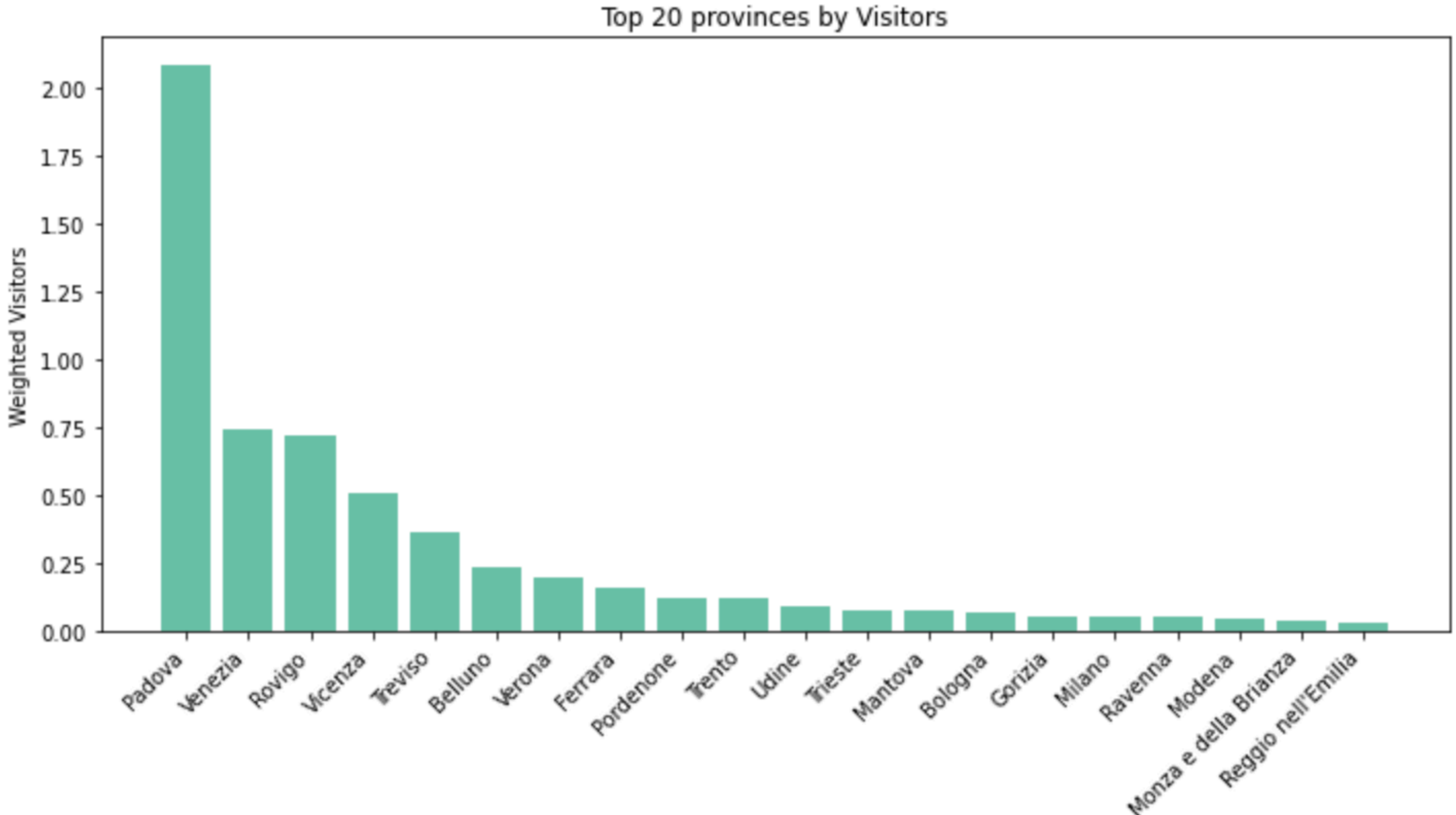
```
#merge with the population dataframe according to the name of province, calculate weighted number of visitors

province_visitors = top_italian_visitors_df.merge(final_population_df[['PROVINCIA', 'POPULATION']],
                                                  on = 'PROVINCIA', how = 'left')

province_visitors['WEIGHTED_VISITORS'] = province_visitors['VISITORS'] / province_visitors['POPULATION']

#sort and form top-20
province_visitors_top_df = province_visitors.sort_values(by='WEIGHTED_VISITORS', ascending=False).head(20)


#preserve only interesting columns
province_visitors_top_df = province_visitors_top_df.drop(columns=['COD_REG','PROV_SIGLA'])
province_visitors_top_df
```

4. Plot dataframe:

```
plt.figure(figsize=(10, 6))
plt.bar(province_visitors_top_df['PROVINCIA'], province_visitors_top_df['WEIGHTED_VISITORS'], color='#66c2a5')
plt.xlabel('Province')
plt.ylabel('Total Visitors')
plt.title('Top 20 provinces by Visitors')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
```

1.2 Ranking of Italian visitors by province, weighted by the number of inhabitants

5. Analyze the results:



2. Study of the visitors' fluxes

1. Create a dataframe consisting of provinces and their directions

```
#create a dictionary according to the province position at the highways
positions = {'south'      : ['Bologna', 'Ferrara', 'Rovigo'],
             'west'       : ['Torino', 'Vercelli', 'Novara', 'Milano', 'Monza e della Brianza',
                             'Bergamo', 'Brescia', 'Verona', 'Vicenza'],
             'north-east' : ['Venezia', 'Treviso', 'Udine', 'Gorizia', 'Trieste']}
```

```
#create a dataframe from dictionary
provinces = [province for province_list in positions.values() for province in province_list]
destinations = [direction for direction in positions.keys() for _ in range(len(positions[direction]))]

positions_df = pd.DataFrame({'PROVINCIA': provinces, 'DIRECTION': destinations})
positions_df
```

2. Study of the visitors' fluxes

2. Create a dataframe consisting of weighted number of visitors per direction

```
#merge the final dataframe from 1.2 with destinations dataframe

province_directions_df = province_visitors_top_df.merge(positions_df[['PROVINCIA','DIRECTION']],
                                                        on = 'PROVINCIA', how = 'left')
```

```
#group datagrame according to number of visitors per each destination, get rid of not interested columns
province_directions_df = province_directions_df.groupby('DIRECTION').sum().drop(columns =
                                                    ['COD_PRO','WEIGHTED_VISITORS'])
```

```
#calculate weighted number of visitors according to the population value per each destination
province_directions_df['WEIGHTED_PER_D'] = province_directions_df['VISITORS'] / province_directions_df['POPULATION']
province_directions_df
```

2. Study of the visitors' fluxes

3. Analyze the results:

DIRECTION	VISITORS	POPULATION	WEIGHTED_PER_D
north-east	1010752	2592397.0	0.389891
south	290276	1577554.0	0.184004
west	821584	5865829.0	0.140063

According to the results obtained, destination north-east (A4 towards Venice - Trieste) should be considered as prioritized direction.

Conclusion

In this project the analysis of Vodafone users' fluxes in Padova from February to March 2018 has been done. Results obtained could provide us with the prioritized directions for future telecommunication resources allocation.

Resources

List of the references and sources we consulted during the project:

- SCWP 2022 Lectures Alberto Zucchetta
- Official python documentation https://docs.python.org/3/
- Official website ISTAT https://www.istat.it/en/analysis-and-products/databases

Thank you for your attention

14.09.2023 | DARIA NIKOLAEVA