

Week-end Challenge - Covid et Mobilité

Annexe Data cleaning / Data preparation.

Dataset bike preparation

```
In [31]: #packages used
import pandas as pd
pd.set_option("display.max_rows",None)
import re
import numpy as np
from pyttrends.request import TrendReq
import pyttrends
import io
import requests

In [4]: df_velo_2019 = pd.read_csv("2019_comptage_velo.csv", sep = ";")
df_velo_2020 = pd.read_csv("2020_comptage_velo_old.csv", sep = ";")

In [5]: df_velo_2019 = df_velo_2019.drop(columns = ["Identifiant du site de comptage",
"Nom du site de comptage",
"Date d'installation du site de comptage",
"Lien vers photo du site de comptage"])

df_velo_2020 = df_velo_2020.drop(columns = ["Identifiant du site de comptage",
"Nom du site de comptage",
"Date d'installation du site de comptage",
"Lien vers photo du site de comptage"])

In [7]: df_velo_2019 = df_velo_2019.rename(columns = {"Identifiant du compteur":"ID_compteur",
"Nom du site de comptage":"location_compteur",
"Date et heure de comptage":"Date_comptage",
"Coordonnées géographiques":"lat_long",
"Comptage horaire":"comptage"})

df_velo_2020 = df_velo_2020.rename(columns = {"Identifiant du compteur":"ID_compteur",
"Nom du compteur":"location_compteur",
"Date et heure de comptage":"Date_comptage",
"Coordonnées géographiques":"lat_long",
"Comptage horaire":"comptage"})

display(df_velo_2019.head())
display(df_velo_2020.head())
```

	ID_compteur	location_compteur	comptage	Date_comptage	lat_long
0	100047539-SC	21 Boulevard Auguste Blanqui SO-NE	0	2019-01-01T07:00:00+01:00	48.830449,2.353199
1	100047539-SC	21 Boulevard Auguste Blanqui SO-NE	31	2019-01-02T18:00:00+01:00	48.830449,2.353199
2	100047539-SC	21 Boulevard Auguste Blanqui SO-NE	24	2019-01-03T16:00:00+01:00	48.830449,2.353199
3	100047539-SC	21 Boulevard Auguste Blanqui SO-NE	2	2019-01-04T04:00:00+01:00	48.830449,2.353199
4	100047539-SC	21 Boulevard Auguste Blanqui SO-NE	0	2019-01-04T08:00:00+01:00	48.830449,2.353199

	ID_compteur	location_compteur	comptage	Date_comptage	lat_long
0	NaN	97 avenue Denfert Rochereau SO-NE	2.0	2019-12-01T04:00:00+01:00	NaN
1	NaN	97 avenue Denfert Rochereau SO-NE	0.0	2019-12-01T03:00:00+01:00	NaN
2	NaN	97 avenue Denfert Rochereau SO-NE	0.0	2019-12-01T05:00:00+01:00	NaN
3	NaN	97 avenue Denfert Rochereau SO-NE	2.0	2019-12-01T08:00:00+01:00	NaN
4	NaN	97 avenue Denfert Rochereau SO-NE	0.0	2019-12-01T06:00:00+01:00	NaN

```
In [10]: def deal_with_date(df) :
    "remove useless information in date columns"
    df = df.astype(str).str.split("T")
    year_month_day = []
    year_month_day = [date[0] for date in df]
    return year_month_day

def sub_number(df):
    """ remove number from string"""
    new_location = []
    for location in df :
        location = re.sub("d+", " ", location)
        new_location.append(location)
    liste_location = new_location
    return liste_location

def sub_card_direction(df):
    """ remove cardinal direction
in location"""
    new_liste = []
    for row in df :
        row = re.sub("([SEON-]+)", " ", row)
        row = row.strip().lower()
        new_liste.append(row)
    return new_liste
```

```
In [11]: #date 2019
date_velo_2019 = deal_with_date(df_velo_2019["Date_comptage"])
df_velo_2019["Date_comptage"] = date_velo_2019
df_velo_2019["Date_comptage"] = pd.to_datetime(df_velo_2019["Date_comptage"])

#date 2020
date_velo_2020 = deal_with_date(df_velo_2020["Date_comptage"])
df_velo_2020["Date_comptage"] = date_velo_2020
df_velo_2020["Date_comptage"] = pd.to_datetime(df_velo_2020["Date_comptage"])
```

```
In [15]: #select only year 2020 within df_velo_2020
year_2020 = df_velo_2020["Date_comptage"].dt.year == 2020
df_velo_2020 = df_velo_2020.loc[year_2020]
```

```
In [16]: #location compteur 2019
clean_loc_2019_1 = sub_number(df_velo_2019["location_compteur"])
df_velo_2019["location_compteur"] = clean_loc_2019_1
clean_loc_2019_2 = sub_card_direction(df_velo_2019["location_compteur"])
df_velo_2019["location_compteur"] = clean_loc_2019_2

#location compteur 2020
clean_loc_2020_1 = sub_number(df_velo_2020["location_compteur"])
df_velo_2020["location_compteur"] = clean_loc_2020_1
clean_loc_2020_2 = sub_card_direction(df_velo_2020["location_compteur"])
df_velo_2020["location_compteur"] = clean_loc_2020_2
```

```
In [19]: #aggregate data "comptage" 2019
clean_df_velo_2019 = df_velo_2019.groupby(["Date_comptage", "location_compteur", "lat_long"])["comptage"].sum()
clean_df_velo_2019 = clean_df_velo_2019.reset_index()

#aggregate data "comptage" 2020
clean_df_velo_2020 = df_velo_2020.groupby(["Date_comptage", "location_compteur", "lat_long"])["comptage"].sum()
clean_df_velo_2020 = clean_df_velo_2020.reset_index()
```

```
In [21]: display(clean_df_velo_2019.head())
display(clean_df_velo_2020.head())
```

	Date_comptage	location_compteur	lat_long	comptage
0	2019-01-01	avenue daumesnil	48.843435,2.383378	350
1	2019-01-01	avenue de la grande armée	48.87451,2.29215	155
2	2019-01-01	avenue de la grande armée	48.874716,2.292439	124
3	2019-01-01	avenue de la porte des ternes	48.8818100845,2.28154603106	75
4	2019-01-01	avenue denfert rochereau	48.834695,2.332968	140

	Date_comptage	location_compteur	lat_long	comptage
0	2020-01-01	avenue d'Italie	48.82024,2.35902	560.0
1	2020-01-01	avenue d'Italie	48.82026,2.3592	572.0
2	2020-01-01	avenue daumesnil	48.843435,2.383378	432.0
3	2020-01-01	avenue de flandre	48.889046,2.374872	977.0
4	2020-01-01	avenue de flandre	48.88926,2.37472	986.0

```
In [22]: #combine dataset 2019 & 2020
df_final_velo = pd.concat([clean_df_velo_2019, clean_df_velo_2020])
```

```
In [24]: #standardise coordinate
mapping = df_final_velo[["location_compteur", "lat_long"]]
mapping = mapping.drop_duplicates("location_compteur")
location = list(mapping["location_compteur"])
lat_long = list(mapping["lat_long"])
dicti_to_map = {loc : lat for loc, lat in zip(location, lat_long)}
df_final_velo["lat_long"] = df_final_velo["location_compteur"].map(dicti_to_map)
```

```
In [27]: df_final_velo.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 38050 entries, 0 to 23986
Data columns (total 4 columns):
#   Column                Non-Null Count  Dtype
---  ---                -
0   Date_comptage         38050 non-null  datetime64[ns]
1   location_compteur     38050 non-null  object
2   lat_long              38050 non-null  object
3   comptage              38050 non-null  float64
dtypes: datetime64[ns](1), float64(1), object(2)
memory usage: 1.5+ MB
```

```
In [28]: #final group by after coordinate have been standardized
df_final_velo = df_final_velo.groupby(["Date_comptage", "location_compteur", "lat_long"])["comptage"].sum()
df_final_velo = df_final_velo.reset_index()
```

```
In [67]: #create global bike traffic 2019
macro_2019 = df_final_velo.groupby(["Date_comptage"])["comptage"].sum()
macro_2019 = macro_2019.reset_index()

#create global bike traffic 2020
macro_2020 = df_final_velo.groupby(["Date_comptage"])["comptage"].sum()
macro_2020 = macro_2020.reset_index()
```

```
In [56]: #mask 1st and second lockdown
date_conf_1 = (df_final_velo["Date_comptage"] > "2020-03-16") & (df_final_velo["Date_comptage"] < "2020-06-02")
date_conf_2 = (df_final_velo["Date_comptage"] > "2020-10-28") & (df_final_velo["Date_comptage"] < "2020-12-31")
```

```
In [57]: df_final_velo.loc[date_conf_1, 'period_conf'] = 1
df_final_velo.loc[date_conf_2, 'period_conf'] = 1
```

```
In [69]: df_final_velo.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 29588 entries, 0 to 29587
Data columns (total 5 columns):
#   Column                Non-Null Count  Dtype
---  ---                -
0   Date_comptage         29588 non-null  datetime64[ns]
1   location_compteur     29588 non-null  object
2   lat_long              29588 non-null  object
3   comptage              29588 non-null  float64
4   period_conf           7220 non-null   float64
dtypes: datetime64[ns](1), float64(2), object(2)
memory usage: 1.1+ MB
```

dataset covid preparation

```
In [32]: df_covid = pd.read_csv("covid_paris.csv")
```

```
In [33]: df_covid.head()
```

```
Out[33]:
```

	Unnamed: 0	date	maille_nom	deces	reanimation	hospitalises	nouvelles_hospitalisations	nouvelles_reanimations	gueris
0	0	2020-03-05	Paris	0.0	0.0	0.0	0.0	0.0	0.0
1	1	2020-03-06	Paris	0.0	0.0	0.0	0.0	0.0	0.0
2	2	2020-03-07	Paris	0.0	0.0	0.0	0.0	0.0	0.0
3	3	2020-03-08	Paris	0.0	0.0	0.0	0.0	0.0	0.0
4	4	2020-03-09	Paris	0.0	0.0	0.0	0.0	0.0	0.0

```
In [36]: def var_day(df) :
    """create a list that contains the % var"""
    liste_var = []
    series = pd.Series(df)
    liste_var.append(series.pct_change()*100)
    return liste_var

def var_abs(df_cumul, df_evo) :
    """find the number between cumulative number
ex 14,22 => 8 """
    return (df_cumul - (df_cumul/(100 + df_evo))*100).round()
```

```
In [38]: var_deces = var_day(df_covid["deces"])
var_rea = var_day(df_covid["reanimation"])
var_hospitalises = var_day(df_covid["hospitalises"])
```

```
In [42]: df_covid["evolution_deces"] = var_deces[0]
df_covid["evolution_rea"] = var_rea[0]
df_covid["evolution_hospitalises"] = var_hospitalises[0]
```

```
In [43]: df_covid = df_covid.replace([np.inf, -np.inf], 0)
df_covid = df_covid.fillna(0)
```

```
In [48]: df_covid["deces_nb"] = var_abs(df_covid["deces"], df_covid["evolution_deces"])
df_covid["rea_nb"] = var_abs(df_covid["reanimation"], df_covid["evolution_rea"])
df_covid["hosp_nb"] = var_abs(df_covid["hospitalises"], df_covid["evolution_hospitalises"])
```

```
In [50]: df_covid.head(10)

#4th dataset that contains covid data
df_covid.to_csv("covid.csv")
```

Dataset metro Paris

```
In [55]: df_metro = pd.read_csv("frequantation_metro_defense.csv", sep=";")
df_metro.head()
```

```
Out[55]:
```

	Date	Type_Jour	Total
0	2019-03-16	SA	171004
1	2019-03-18	JOHV	366748
2	2019-03-21	JOHV	378217
3	2019-03-26	JOHV	383891
4	2019-03-29	JOHV	379865

```
In [60]: date_conf_1 = (df_metro["Date"] > "2020-03-16") & (df_metro["Date"] < "2020-06-02")
date_conf_2 = (df_metro["Date"] > "2020-10-28") & (df_metro["Date"] < "2020-12-31")
```

```
In [61]: df_metro.loc[date_conf_1, 'period'] = 1
df_metro.loc[date_conf_2, 'period'] = 1
```

```
In [62]: #5th data set on La Defense traffic
df_metro.to_csv("metro_1.csv")
```

Dataset velib Paris

```
In [63]: df_velib = pd.read_csv("velib.csv", sep=";")
```

```
In [64]: df_velib_less = df_velib[["Capacité de la station", "Vélos mécaniques disponibles", "Vélos électriques disponibles"],
"Coordonnées géographiques",
"Nom communes équipées", "Station en fonctionnement"]]

df_velib_less = df_velib_less.loc[df_velib_less["Nom communes équipées"] == "Paris"]
```

```
In [65]: #6th data set on velib
df_velib_less.to_csv("velib_clean.csv")
```

Dataset google trend

```
In [ ]: searches = ["velo electrique", "velib", "velib paris", "borne velib", "velo", "velo electrique prime", "velo electrique subvention", "velo tout chemin", "piste cyclable", "velo decathlon", "achat de velo"]
```

```
In [ ]: def search_gtrend_df(l) :
    """use a list to search words on G trends
and return a df with date, search and the trend score"""

    pytrend = TrendReq()

    dicti = {}
    for trending in l :
        pytrend.build_payload([trending], timeframe = '2018-12-31 2020-12-31', geo = "FR-J")
        dicti[trending] = pytrend.interest_over_time()
```

```
In [ ]: df = pd.concat(dicti, axis=1)
df.columns = df.columns.droplevel(0)
df = df.drop('isPartial', axis = 1)
df = df.reset_index()
df = pd.melt(df, id_vars='date', value_vars=1)

return df
```

```
In [ ]: df_gtrends = search_gtrend_df(searches)
df_gtrends.info()
```

```
In [ ]: df_gtrends.to_csv("google_trend.csv", index=False)
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```