Treatment

March 4, 2024

[ ]:
```
import pandas as pd
import numpy as np
import random
dataset=pd.read_csv("example/id_3.csv")
datamod=dataset.copy()
print(pd.__version__)
print(np.__version__)
```

0.0.1 Times series convertion

[ ]:
```
serie = pd.Series(np.array(dataset['Daily Mean PM2.5 Concentration']), index =␣
 ↪dataset['Date'])
serie.index = pd.to_datetime(serie.index, format='%m/%d/%Y', errors='coerce')
serie.sort_index(inplace=True)      # important to apply loc,  loc need␣
 ↪corrected sort

##for datasets
dataset = dataset.dropna()

data_unique = dataset.drop_duplicates(implace=True)
print(f'DataFrame original shape: {data_original.shape}')
print(f'data unique shape: {data_unique.shape}')
```

0.0.2 temporal resize

[ ]:
```
def resize_data_mod(time_freq):
    global data_mod

    reset_data_mod()

    data_mod=data_mod.resample(time_freq).mean()
    data_mod = data_mod.fillna(data_mod.mean())
# resize_data_mod("30D")
data_mod.head(50)
 ##                 orrrrr
 data_mod_resampled = data_mod.resample(time_freq).mean().dropna()  #
 ##      ´spline´ for temporal is the best , 'linear'   low variation   ,
```

```
##  'Nearest Neighbors'   high variation, crazy data, low precion result, but␣
 ↪best than other in this case
##     'Polinomial '  low rate, need adust degree
```

0.0.3 others

[ ]:
```
print(type(serie))
print(type(serie.index))
print(serie.dtype)
print(serie.ndim)
print(serie.size)
print(serie.info)
```

[ ]:
```
print (serie.shape)
print (serie.describe)
print(dataset.columns)      #only data frames
```

[ ]:
```
print(serie.iloc[1999-1-1:1999-1-4])   ## "iloc doesn't use 0 in front, and the␣
 ↪index needs to be in a date format."
print(serie.loc[serie.index < "2000-08"])
```

[ ]:
```
serie.sum()
serie.mean()
serie.min()
serie.max()
serie.loc[(serie.index >= "2000-01") & (serie.index < "2002-01")].sum()
serie.loc[(pd.DatetimeIndex(serie.index).month == 7)].sum()
datamod["income"].value_counts()      #mostar total de cada valor da coluna
```

[ ]:
```
serie.index.unique
serie.isna().sum()
pd.DatetimeIndex(serie.index).year
```

[ ]:
```
datamod.rename(columns={'c#default':'debtor'}, inplace=True)
datamod['debtor'] = datamod['debtor'].replace({1: True, 0: False})
datamod['concatenated'] = datamod['column1'].astype(str) + datamod['column2']
datamod.drop(columns=['column1','column2'], inplace=True)
datamod_doubleframe = pd.concat([datamod1, datamod2])
```