W12_datapreprocessing

January 12, 2021

1 Dataframe consumption to put into different models

This notebook is used to load and scale data to use in different models

>Jefry
el
Bh-
wash
16095065
>Niels
van
Schaik
18150845
>Amin
Man-
souri
18097367

- 24 hours
- 12 hours
- 6 hours
- 4 hours
- 1 hour

[1]:
```
samplehours = 1
```

[2]:
```
import pandas as pd
from sklearn.preprocessing import StandardScaler
import numpy as np

import sys
sys.path.insert(0,'/home/16095065/notebooks/zero/Imports:/')
import Load_data as ld #resample.mean()
a_dict = {'energyHeatpump': [2],
         'smartMeter': [6,7],
         'solar' : [2,3]}
```

2 Loading Data

```
[3]: savepath = '/home/16095065/notebooks/zero/Data:/modelData/'
     #Reading data from numpy files
     house = ld.load(28,28,a_dict,16095065)
     df = house[28]
```

3 Calculating consumption

Without heatpump this time

- Everything get's changed to show the delta instead of the actual value
- Only consumption remains in the dataframe

  $Consumption = Energy_{out} + (Solar_{in} - Solar_{out}) - Energy_{out}$

```
[4]: #Calculating consumption
     df = df.diff()
     df['consumption'] = df.apply(lambda x: x['smartMeter_6'] +␣
      ↪(x['solar_3']-x['solar_2'])-x['smartMeter_7'], axis=1)
     df = df.dropna()
     df = df.filter(['consumption'])
     df.head(1)
```

```
[4]:                      consumption
     2018-12-31 23:05:00        0.052
```

3.0.1 Resampling

```
[5]: # Resampling to an hour
     df = df['2019']
     df = df.resample(str(samplehours)+'H').sum()
```

```
[6]: df.head(2)
```

```
[6]:                      consumption
     2019-01-01 00:00:00        0.402
     2019-01-01 01:00:00        0.264
```

4 Adding Features to consumption

Shifted consumption values are added now, the hour of the measurement is added only after scaling the data

4.1 Shifting consumption

[7]:
```python
def shifting(sf, shift:int):
    sf['cons_T-'+str(shift)] = sf['consumption'].shift(periods=shift, freq='H')
    return sf

temp_df = df.filter(items=['consumption'])
day_temp_df = df.filter(items=['consumption'])

shiftDagen = [24, 48, 72, 168]

#week
for i in range(24, 168+1):
    temp_df = shifting(temp_df, i)
temp_df = temp_df.drop(['consumption'], axis=1)

#day
for i in range(24, 48+1):
    day_temp_df = shifting(day_temp_df, i)
day_temp_df = day_temp_df.drop(['consumption'], axis=1)

#Shifted days
for i in shiftDagen:
    df = shifting(df, i)

#columns added
df['day_mean'] = day_temp_df.mean(axis=1, skipna=True)
df['week_mean'] = temp_df.mean(axis=1, skipna=True)

df = df.fillna(0)
df.tail(3)
```

[7]:
```
                     consumption  cons_T-24  cons_T-48  cons_T-72  cons_T-168  \
2019-12-31 20:00:00        1.315      0.883      1.047      1.198       0.880
2019-12-31 21:00:00        0.701      0.696      1.163      1.106       0.568
2019-12-31 22:00:00        1.179      0.742      1.010      1.017       0.662

                     day_mean  week_mean
2019-12-31 20:00:00   0.95528   0.911345
2019-12-31 21:00:00   0.94124   0.910076
2019-12-31 22:00:00   0.92440   0.911276
```

5 Adding Hour of the measurement

[8]:
```
# #Adding hour one hot encoded
# hf = pd.DataFrame(index=df.index)
# hf['hour'] = df.index.hour
# hf = pd.get_dummies(hf['hour'], prefix='hour')

# #Merging
# df = pd.merge(df, hf, left_index=True, right_on=hf.index)
# df = df.drop('key_0', axis=1)
```

6 Saving the Result

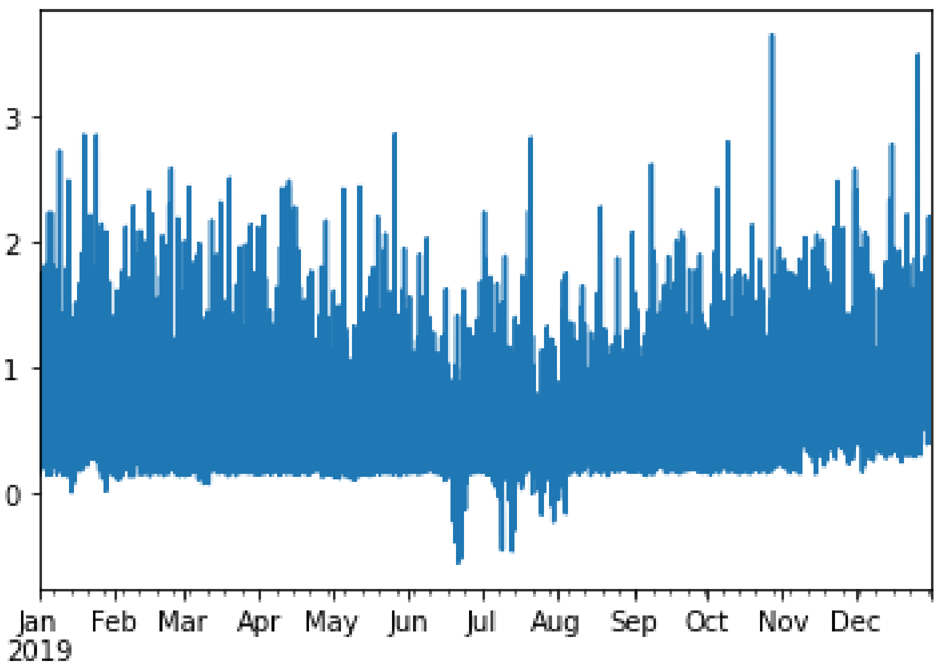
[9]:
```
#print(df.head(10))
# df.dtypes
```

```
                     consumption  cons_T-24  cons_T-48  cons_T-72  cons_T-168  \
2019-01-01 00:00:00        0.402        0.0        0.0        0.0         0.0
2019-01-01 01:00:00        0.264        0.0        0.0        0.0         0.0
2019-01-01 02:00:00        0.163        0.0        0.0        0.0         0.0
2019-01-01 03:00:00        1.526        0.0        0.0        0.0         0.0
2019-01-01 04:00:00        0.274        0.0        0.0        0.0         0.0
2019-01-01 05:00:00        0.799        0.0        0.0        0.0         0.0
2019-01-01 06:00:00        0.733        0.0        0.0        0.0         0.0
2019-01-01 07:00:00        1.041        0.0        0.0        0.0         0.0
2019-01-01 08:00:00        0.255        0.0        0.0        0.0         0.0
2019-01-01 09:00:00        0.152        0.0        0.0        0.0         0.0

                     day_mean  week_mean
2019-01-01 00:00:00       0.0        0.0
2019-01-01 01:00:00       0.0        0.0
2019-01-01 02:00:00       0.0        0.0
2019-01-01 03:00:00       0.0        0.0
2019-01-01 04:00:00       0.0        0.0
2019-01-01 05:00:00       0.0        0.0
2019-01-01 06:00:00       0.0        0.0
2019-01-01 07:00:00       0.0        0.0
2019-01-01 08:00:00       0.0        0.0
2019-01-01 09:00:00       0.0        0.0
```

[10]:
```
#df['consumption'].plot()
```

[10]:
```
<AxesSubplot:>
```



```
[11]: #df.to_pickle(str(savepath)+'_v01_'+str(samplehours))
```