W09_MVLR

January 12, 2021

1 MVLR: with a new dataset

Niels van Drunen 18062814

Jefry el Bhwash 16095065

[1]:

```
#modules
import numpy as np
import pandas as pd
from tqdm import tqdm
import matplotlib.pyplot as plt
import glob
from sklearn.metrics import r2_score
import sklearn
from sklearn.model_selection import train_test_split
from sklearn.model_selection import cross_validate
import seaborn as sns
```

1.0.1 Import of data

[2]:

```
loadpath = '/home/16095065/notebooks/zero/datasetP/'
greathouses = [37,40,41,42,51,53,54,55,56,57,58,60,70,72,99,100,105,108,114,115]
houses = {}
for h in greathouses:
    houses[h] = pd.read_pickle(loadpath + 'Train_' +str(h)).fillna(0)
```

[3]:

```
houses[37].head()
```

[3]:

```
                     s_delta  solar_T-24  solar_T-48  solar_T-72  \
DateTime
2019-01-01 00:00:00      0.0         0.0         0.0         0.0
2019-01-01 01:00:00      0.0         0.0         0.0         0.0
2019-01-01 02:00:00      0.0         0.0         0.0         0.0
2019-01-01 03:00:00      0.0         0.0         0.0         0.0
2019-01-01 04:00:00      0.0         0.0         0.0         0.0

                     straling_T-24  straling_T-48  straling_T-72  \
DateTime
2019-01-01 00:00:00            0.0            0.0            0.0
2019-01-01 01:00:00            0.0            0.0            0.0
2019-01-01 02:00:00            0.0            0.0            0.0
2019-01-01 03:00:00            0.0            0.0            0.0
2019-01-01 04:00:00            0.0            0.0            0.0

                     temperature_T-24  temperature_T-48  temperature_T-72
DateTime
2019-01-01 00:00:00              93.0               0.0               0.0
2019-01-01 01:00:00              95.0               0.0               0.0
2019-01-01 02:00:00              92.0               0.0               0.0
2019-01-01 03:00:00              90.0               0.0               0.0
2019-01-01 04:00:00              90.0               0.0               0.0
```

2 Data cleaning

[4]:

```
h=37
#houses[h].apply(lambda x: if houses[h][s_delta] > 10, then fillna)
#houses[h] = houses[h]['s_delta'].apply(lambda x: 0 if x > 10.0 else x)
#houses[h].loc[houses[h]['s_delta'] >8, 's_delta'] = 0#houses[h]['s_delta'].
 ↪mean()
houses[h].head()
```

[4]:

```
                     s_delta  solar_T-24  solar_T-48  solar_T-72  \
DateTime
2019-01-01 00:00:00      0.0         0.0         0.0         0.0
2019-01-01 01:00:00      0.0         0.0         0.0         0.0
2019-01-01 02:00:00      0.0         0.0         0.0         0.0
2019-01-01 03:00:00      0.0         0.0         0.0         0.0
2019-01-01 04:00:00      0.0         0.0         0.0         0.0

                     straling_T-24  straling_T-48  straling_T-72  \
DateTime
2019-01-01 00:00:00            0.0            0.0            0.0
2019-01-01 01:00:00            0.0            0.0            0.0
2019-01-01 02:00:00            0.0            0.0            0.0
2019-01-01 03:00:00            0.0            0.0            0.0
2019-01-01 04:00:00            0.0            0.0            0.0

                     temperature_T-24  temperature_T-48  temperature_T-72
DateTime
2019-01-01 00:00:00              93.0               0.0               0.0
2019-01-01 01:00:00              95.0               0.0               0.0
2019-01-01 02:00:00              92.0               0.0               0.0
```

```
2019-01-01 03:00:00              90.0              0.0              0.0
2019-01-01 04:00:00              90.0              0.0              0.0
```

3 Model

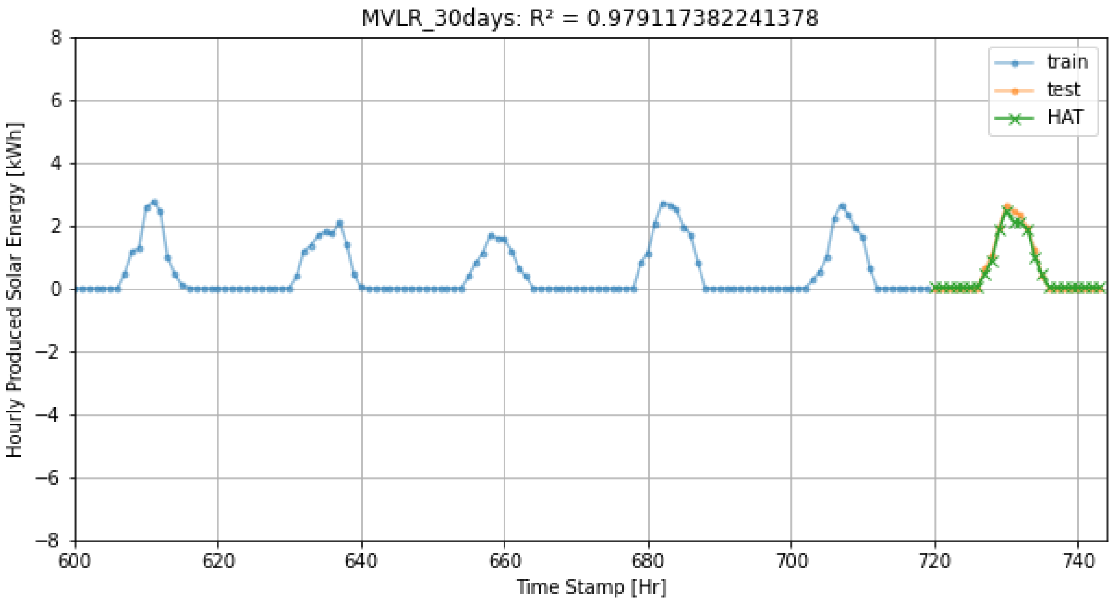
[5]:

```
%matplotlib inline
from sklearn import linear_model
days = 1

df = houses[37]
df = df['2019-10-01':'2019-10-31']
df['hour'] = df.index.hour

features = ['solar_T-24','solar_T-48', 'solar_T-72', 'straling_T-24',␣
 ↪'straling_T-48', 'straling_T-72', 'hour'] #, 'temperature_T-24',␣
 ↪'temperature_T-48', 'temperature_T-72'
target = 's_delta'

X = df[features].values.reshape(-1, len(features))
y = df[target].values
y = y.reshape(y.shape[0], 1)
print(X.shape)
print(y.shape)
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=(1/len(df.
 ↪index)*24)*days, random_state=0,shuffle=False)

regr = linear_model.LinearRegression()
regr.fit(X_train,y_train)

print('Intercept: \n', regr.intercept_)
print('Coefficients: \n', regr.coef_)
y_hat =  regr.predict(X_test)

plt.figure(figsize=(10,5))
plt.plot(np.arange(X_train.shape[0]), y_train, ".-", label='train', alpha=0.5)
plt.plot(np.arange(X_train.shape[0], X_train.shape[0]+X_test.shape[0]), y_test,␣
 ↪ ".-", label='test', alpha=0.5)
plt.plot(np.arange(X_train.shape[0], X_train.shape[0]+X_test.shape[0]), y_hat, ␣
 ↪ "x-", label='HAT')
plt.xlabel('Time Stamp [Hr]')
plt.ylabel('Hourly Produced Solar Energy [kWh]')
plt.title('MVLR_30days: R\u00b2 = ' + str(r2_score(y_hat, y_test)))

plt.ylim([-8,8])
plt.xlim([X_train.shape[0]-(24*5),X_train.shape[0]+X_test.shape[0]]) #lastpart
```

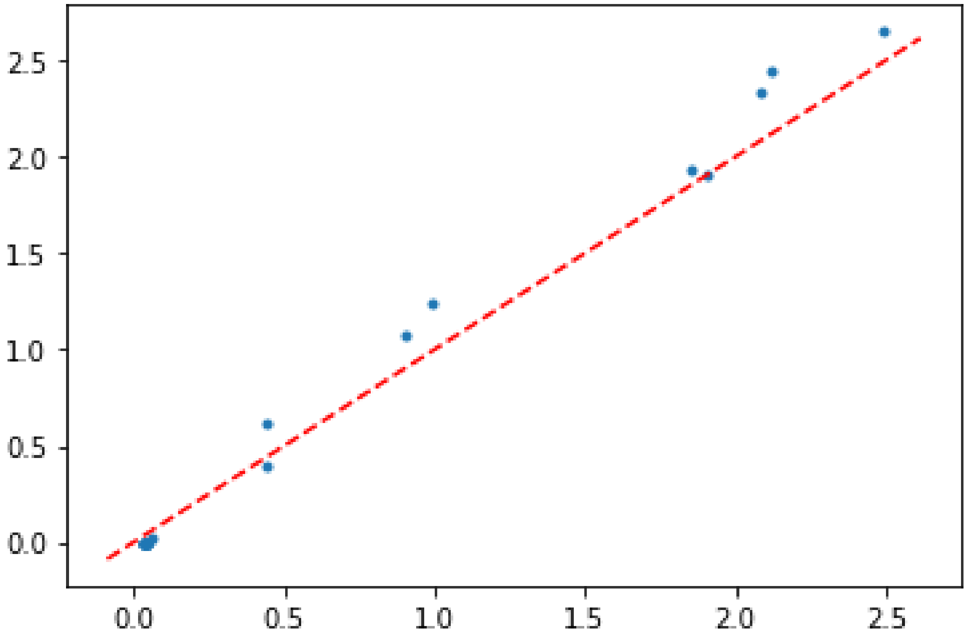
```
#plt.xlim([0,X_train.shape[0]+X_test.shape[0]]) #year
plt.grid()
plt.legend()
## R^2 functie toepassen op yhat vs y
print('R\u00b2 score: ', r2_score(y_hat, y_test))
plt.savefig('W9_MVLR_month.png', dpi=600)
```

```
/opt/jupyterhub/anaconda/lib/python3.6/site-packages/ipykernel_launcher.py:7:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  import sys
```

```
(744, 7)
(744, 1)
Intercept:
 [0.03542919]
Coefficients:
 [[-1.71410383e-03  4.03012484e-02  4.11921996e-01  7.62574130e-03
   6.46424598e-03 -4.88276609e-03  1.44996010e-04]]
R² score:  0.979117382241378
```



[6]:
```
plt.plot(y_hat, y_test, ".")
plt.plot(plt.xlim(), plt.xlim(), ls="--", c='r', label="$y$=$\hat{y}$")
plt.savefig('W9_MVLR_month_y_yhat.png', dpi=600)
```



```
[7]: cv_results = cross_validate(regr, X, y, cv=2)
     print(cv_results)
```

```
{'fit_time': array([0.00101304, 0.00083661]), 'score_time': array([0.00074959,
0.00052667]), 'test_score': array([0.57463938, 0.76253793])}
```

```
[ ]:
```

```
[ ]:
```