W11_SVR_Consumption

January 12, 2021

1 SVR model Energy Consumption

Een SVR model ter referentie voor bij het Multi Layer Perceptron model.

Niels van Drunen

Levy Duivenvoorden

Amin Mansouri

1.0.1 Importeer modules

[56]:

```
#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
from sklearn import linear_model
from sklearn.svm import SVR
from sklearn.preprocessing import StandardScaler
```

1.1 Data klaarmaken

[57]:

```
%matplotlib notebook
df = pd.read_pickle('28_df_consumption_weekday')

#scale the data
scaler = StandardScaler()
scaler.fit(df.loc[:,~df.columns.isin(["consumption"])])
scaled_dataX = scaler.transform(df.loc[:,~df.columns.isin(["consumption"])]).
 ↪tolist()
```

```
datay = np.array(df["consumption"].tolist())

#split the data
train_X = np.array(scaled_dataX[0:24*9])
train_y = datay[0:24*9].reshape(-1,1)

valid_X = np.array(scaled_dataX[24*9:24*10])
valid_y = datay[24*9:24*10].reshape(-1,1)

## sccaler op train fitten
# test_X = scaled_dataX[24*10:24*11]
# test_y = datay[24*10:24*11].reshape(-1,1)
# scaler op test fitten
```

[58]:

```
df.head()
```

[58]:

```
                     consumption  day_0  day_1  day_2  day_3  day_4  day_5  \
2019-01-01 23:00:00        0.209      0      1      0      0      0      0
2019-01-02 00:00:00        0.253      0      0      1      0      0      0
2019-01-02 01:00:00        0.332      0      0      1      0      0      0
2019-01-02 02:00:00        0.597      0      0      1      0      0      0
2019-01-02 03:00:00        0.825      0      0      1      0      0      0

                     day_6  cons_T-1  cons_T-2  …  cons_T-15  cons_T-16  \
2019-01-01 23:00:00      0     0.381     0.403  …      0.255      1.041
2019-01-02 00:00:00      0     0.209     0.381  …      0.152      0.255
2019-01-02 01:00:00      0     0.253     0.209  …      0.435      0.152
2019-01-02 02:00:00      0     0.332     0.253  …      0.810      0.435
2019-01-02 03:00:00      0     0.597     0.332  …      0.712      0.810

                     cons_T-17  cons_T-18  cons_T-19  cons_T-20  cons_T-21  \
2019-01-01 23:00:00      0.733      0.799      0.274      1.526      0.163
2019-01-02 00:00:00      1.041      0.733      0.799      0.274      1.526
2019-01-02 01:00:00      0.255      1.041      0.733      0.799      0.274
2019-01-02 02:00:00      0.152      0.255      1.041      0.733      0.799
2019-01-02 03:00:00      0.435      0.152      0.255      1.041      0.733

                     cons_T-22  cons_T-23  cons_T-24
2019-01-01 23:00:00      0.264      0.402      0.622
2019-01-02 00:00:00      0.163      0.264      0.402
2019-01-02 01:00:00      1.526      0.163      0.264
2019-01-02 02:00:00      0.274      1.526      0.163
2019-01-02 03:00:00      0.799      0.274      1.526

[5 rows x 32 columns]
```

1.2 SVR Trainen

[59]:

```
#train the model:
svr = SVR(C=1)
svr.fit(train_X,train_y)

#make data plot ready:
valid = np.arange(train_X.shape[0], train_X.shape[0]+valid_X.shape[0])
train = np.arange(train_X.shape[0])
```

```
/opt/jupyterhub/anaconda/lib/python3.7/site-
packages/sklearn/utils/validation.py:72: DataConversionWarning: A column-vector
y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  return f(**kwargs)
```

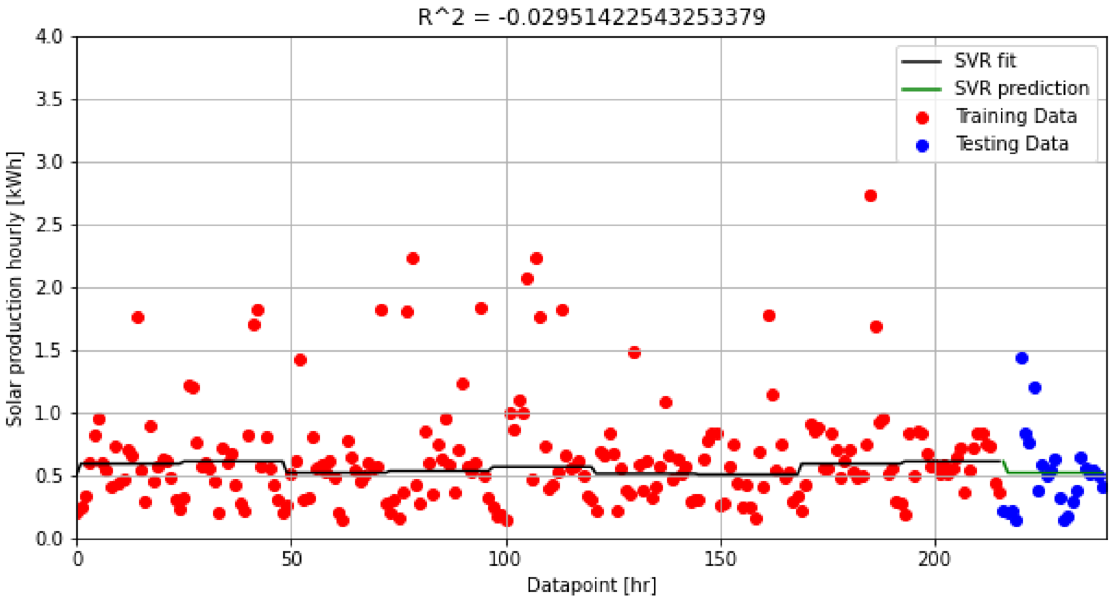
1.2.1 Plotje

[62]:

```
%matplotlib inline
plt.subplots(figsize=(10,5))
plt.plot(train,svr.predict(train_X),color='black',label="SVR fit")
plt.plot(valid,svr.predict(valid_X),color='g',label="SVR prediction")
plt.scatter(train,train_y,color="r",label="Training Data")
plt.scatter(valid,valid_y,color="b",label="Testing Data")

#Nice layout:
plt.xlabel("Datapoint [hr]")
plt.ylabel("Solar production hourly [kWh]")
plt.legend(loc="upper right")
plt.title("R^2 = "+str(svr.score(valid_X,valid_y)))
plt.ylim([0,4])
#plt.xlim([X_train.shape[0]-(24*5),X_train.shape[0]+X_test.shape[0]])
plt.xlim([0,train_X.shape[0]+valid_X.shape[0]])
plt.grid()
plt.show()

#print de score:
print("R2-score test data:")
print(svr.score(valid_X,valid_y))
```



```
R2-score test data:
-0.029514225432533379
```

1.3 Evaluatie

[61]:

```
%matplotlib notebook
plt.scatter(valid_y,svr.predict(valid_X))
plt.xlabel("y [kWh]")
plt.ylabel("y_hat [kWh]")
#plt.legend(loc="upper right")
plt.title("$R^2$ = %.2f" % (r2_score(valid_y,svr.predict(valid_X))))
plt.plot(plt.xlim(), plt.xlim(), ls="--", c='r', label="$y$=$\hat{y}$")
plt.grid()
plt.show()
```

```
<IPython.core.display.Javascript object>
```

```
<IPython.core.display.HTML object>
```

[ ]: