W12_NeuralNetwork-Copy8

January 12, 2021

1 Een simpel Neuraal Netwerk in PyTorch

Dit is gemaakt door: - Levy Duivenvoorden (18005152) - Jefry el Bhwash (16095065) - Amin Mansouri - Niels van Drunen - Niels van Schaik

aangemaakt: 25/11/2020

1.1 importeer alle modules:

[1]:
```
import random
import time
import torch
import torchvision
import torchvision.transforms as transforms
import numpy as np
import pandas as pd
import math
import random

import torch.nn as nn
import torch.nn.functional as F
import torch.optim as optim

import seaborn as sns
import matplotlib.pyplot as plt

from tqdm import tqdm
from IPython.display import clear_output
from sklearn.metrics import r2_score
from sklearn.metrics import mean_squared_error
from sklearn.preprocessing import StandardScaler
from sklearn.feature_selection import SelectKBest, mutual_info_classif
from sklearn.metrics import mean_absolute_error
import wandb

# hier random seeds mee geven
random.seed(1337)
```

```
torch.manual_seed(1337)

%matplotlib inline
%config InlineBackend.print_figure_kwargs={'facecolor' : "w"}
```

[2]:
```
# CUDA initialisation
ngpu = torch.cuda.device_count() # number of available gpus
device = torch.device("cuda:4") if (torch.cuda.is_available() and ngpu > 0)␣
 ↪else "cpu" #cuda:0 for gpu 0, cuda:4 for gpu 5
torch.backends.cudnn.benchmark=True # Uses cudnn auto-tuner to find the best␣
 ↪algorithm to use for your hardware
```

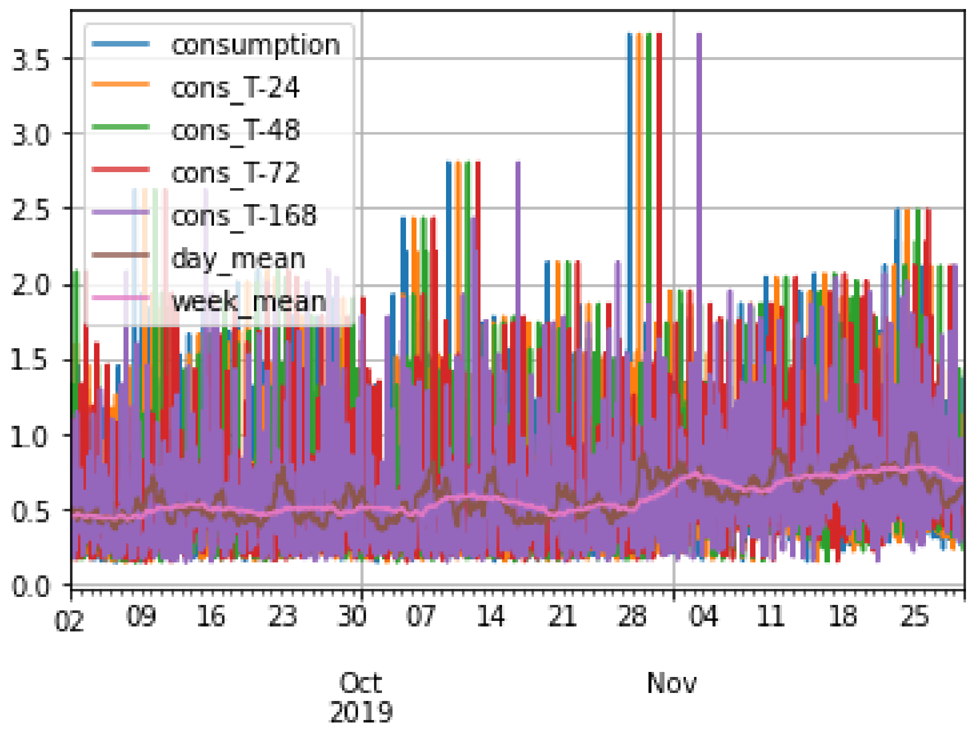
1.2 Laad de dataframe in:

[3]:
```
df = pd.read_pickle('/home/18005152/notebooks/zero/Data:/modelData/_v01_1')
df = df['2019-09-02':'2019-11-29']
```

1.3 Laat de data zien:

[4]:
```
%matplotlib inline
df.plot()
plt.grid()
df.head()
```

[4]:
```
                     consumption  cons_T-24  cons_T-48  cons_T-72  cons_T-168  \
2019-09-02 00:00:00        0.183      0.170     0.1790      0.604       0.227
2019-09-02 01:00:00        0.239      0.171     0.2090      0.563       0.214
2019-09-02 02:00:00        0.290      0.246     1.3335      0.780       1.099
2019-09-02 03:00:00        0.302      0.422     0.5510      1.101       0.768
2019-09-02 04:00:00        0.234      1.188     0.7450      0.485       0.744

                     day_mean  week_mean
2019-09-02 00:00:00   0.51006   0.460259
2019-09-02 01:00:00   0.50974   0.459872
2019-09-02 02:00:00   0.51122   0.460093
2019-09-02 03:00:00   0.47476   0.455424
2019-09-02 04:00:00   0.50024   0.458321
```



1.4 Train Validate Test split (ok)

[5]:
```
# lijst = ["consumption"]
# for i in range(0,24):
#     lijst.append("hour_"+str(i))
# print(lijst)
```

[6]:
```
#scale the data
#X:
scalerx = StandardScaler()
lijst = ["consumption"]
# for i in range(0,24):
#     lijst.append("hour_"+str(i))
scalerx.fit(df.loc[:,~df.columns.isin(lijst)])
scaled_dataX = scalerx.transform(df.loc[:,~df.columns.isin(lijst)]).tolist()
# for j in range(0,24):
#     for i in range(0,len(df["hour_"+str(j)].tolist())):
#         scaled_dataX[i][j] = df["hour_"+str(j)].tolist()[i]

#Y:
scalery = StandardScaler()
scalery.fit(df.loc[:,df.columns.isin(["consumption"])])
```

```
datay = scalery.transform(df.loc[:,df.columns.isin(["consumption"])])

start = 0
week = 7*1*24
end_train = -2*week
end_valid = -1*week
end_test = -1
#split the data
train_X = scaled_dataX[start:end_train]
train_y = datay[start:end_train].reshape(-1,1)

valid_X = scaled_dataX[end_train:end_valid]
valid_y = datay[end_train:end_valid].reshape(-1,1)

test_X = scaled_dataX[end_valid:end_test]
test_y = datay[end_valid:end_test].reshape(-1,1)

#Make tensors from the numpy arrays.
train_X_t = torch.from_numpy(np.array(train_X)).to(device).float()
train_y_t = torch.from_numpy(np.array(train_y)).to(device).float()

valid_X_t = torch.from_numpy(np.array(valid_X)).to(device).float()
valid_y_t = torch.from_numpy(np.array(valid_y)).to(device).float()

test_X_t = torch.from_numpy(np.array(test_X)).to(device).float()
test_y_t = torch.from_numpy(np.array(test_y)).to(device).float()

#Tensor Datasets
train_set = torch.utils.data.TensorDataset(train_X_t, train_y_t)
test_set = torch.utils.data.TensorDataset(valid_y_t, valid_X_t)

#Tensor DataLoaders
train_loader = torch.utils.data.DataLoader(train_set, batch_size=64,␣
 ↪shuffle=False, num_workers = 0)#, pin_memory=True)
test_loader = torch.utils.data.DataLoader(test_set, batch_size=64,␣
 ↪shuffle=False, num_workers = 0)#, pin_memory=True)
```

1.5 Neuraal Netwerk:

[7]:
```
#Parameters:
layerSize = 128
outputSize = 1
featureSize = train_X_t.shape[1]
relu = nn.ReLU()

#class maken voor NN
```

```
class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.fc1 = nn.Linear(featureSize, layerSize)
        self.fc2 = nn.Linear(layerSize, outputSize)

    def forward(self, x):
        x = relu(self.fc1(x))
        x = self.fc2(x)
        return x
```

[8]:
```
#make model:
model = Net().to(device)
model.float()
```

[8]:
```
Net(
  (fc1): Linear(in_features=6, out_features=128, bias=True)
  (fc2): Linear(in_features=128, out_features=1, bias=True)
)
```

1.6 Train het Neuraal Netwerk:

[9]:
```
#Training parameters:
train_for = 1000
show_every = 5
optimizer = optim.Adam(model.parameters(),lr=3e-4)
criterion = nn.MSELoss()

#initialize:
ite = 0
epochs = range(1,train_for)
alijst = []; blijst = []

#Learning loop:
for i in (epochs):
    btime = time.time()
    model.train()
    for batch_idx, data_target in enumerate(train_loader):
        data = data_target[0]
        target = data_target[1]
        data = data.view(-1, data.shape[1])
        optimizer.zero_grad()

        output = model(data)
        loss = criterion(output, target)
        loss.backward()
        optimizer.step()
    etime = time.time()

    model.eval()
    if (ite%show_every) == 0:
        y = valid_y_t.cpu().detach().numpy()
        yhat = model(valid_X_t.float()).cpu().detach().numpy()
        # hier loss appenden ipv nog een losse berekening
        alijst.append(loss)
        blijst.append(mean_absolute_error(y,yhat))

        y = train_y_t.cpu().detach().numpy()
        yhat = model(train_X_t.float()).cpu().detach().numpy()
        # R^2-score:
        r2 = r2_score(yhat, y)
        # RMSE:
        rmse = np.sqrt(mean_squared_error(yhat, y))
        # MAPE:
        actual, pred = np.array(y), np.array(yhat)
        mape = np.mean(np.abs((actual - pred) / actual)) * 100
        # MAE:
        mae = mean_absolute_error(yhat, y)
        #print:
        print('Epoch: %d\t R\u00b2: %.2f\t RMSE: %.2f\t MAPE: %.2f\t MAE: %.
↪2f\t Looptime: %.3f s' % (ite,r2,rmse,mape,mae,etime-btime))
    ite+=1
```

```
Epoch: 0         R²: -19.22      RMSE: 0.94      MAPE: 183.06    MAE: 0.64
Looptime: 0.470 s
Epoch: 5         R²: -3.50       RMSE: 0.88      MAPE: 204.66    MAE: 0.59
Looptime: 0.065 s
Epoch: 10        R²: -2.86       RMSE: 0.88      MAPE: 202.27    MAE: 0.58
Looptime: 0.065 s
Epoch: 15        R²: -2.64       RMSE: 0.87      MAPE: 204.17    MAE: 0.57
Looptime: 0.065 s
Epoch: 20        R²: -2.50       RMSE: 0.87      MAPE: 207.12    MAE: 0.57
Looptime: 0.067 s
Epoch: 25        R²: -2.39       RMSE: 0.86      MAPE: 210.36    MAE: 0.57
Looptime: 0.068 s
Epoch: 30        R²: -2.30       RMSE: 0.86      MAPE: 213.52    MAE: 0.56
Looptime: 0.068 s
Epoch: 35        R²: -2.23       RMSE: 0.86      MAPE: 216.61    MAE: 0.56
Looptime: 0.067 s
Epoch: 40        R²: -2.17       RMSE: 0.86      MAPE: 219.05    MAE: 0.56
Looptime: 0.068 s
Epoch: 45        R²: -2.11       RMSE: 0.86      MAPE: 221.16    MAE: 0.56
Looptime: 0.068 s
```

```
Epoch: 50        R²: -2.07       RMSE: 0.86      MAPE: 222.76    MAE: 0.56
Looptime: 0.068 s
Epoch: 55        R²: -2.03       RMSE: 0.85      MAPE: 223.96    MAE: 0.56
Looptime: 0.068 s
Epoch: 60        R²: -2.00       RMSE: 0.85      MAPE: 224.78    MAE: 0.56
Looptime: 0.065 s
Epoch: 65        R²: -1.97       RMSE: 0.85      MAPE: 225.47    MAE: 0.56
Looptime: 0.065 s
Epoch: 70        R²: -1.95       RMSE: 0.85      MAPE: 225.89    MAE: 0.56
Looptime: 0.076 s
Epoch: 75        R²: -1.93       RMSE: 0.85      MAPE: 226.11    MAE: 0.56
Looptime: 0.068 s
Epoch: 80        R²: -1.91       RMSE: 0.85      MAPE: 226.35    MAE: 0.56
Looptime: 0.067 s
Epoch: 85        R²: -1.90       RMSE: 0.85      MAPE: 226.46    MAE: 0.56
Looptime: 0.068 s
Epoch: 90        R²: -1.89       RMSE: 0.85      MAPE: 226.48    MAE: 0.56
Looptime: 0.068 s
Epoch: 95        R²: -1.87       RMSE: 0.85      MAPE: 226.61    MAE: 0.56
Looptime: 0.067 s
Epoch: 100       R²: -1.86       RMSE: 0.85      MAPE: 226.73    MAE: 0.56
Looptime: 0.068 s
Epoch: 105       R²: -1.85       RMSE: 0.85      MAPE: 226.94    MAE: 0.55
Looptime: 0.068 s
Epoch: 110       R²: -1.84       RMSE: 0.85      MAPE: 227.15    MAE: 0.55
Looptime: 0.068 s
Epoch: 115       R²: -1.83       RMSE: 0.85      MAPE: 227.25    MAE: 0.55
Looptime: 0.069 s
Epoch: 120       R²: -1.82       RMSE: 0.85      MAPE: 227.41    MAE: 0.55
Looptime: 0.068 s
Epoch: 125       R²: -1.81       RMSE: 0.85      MAPE: 227.48    MAE: 0.55
Looptime: 0.070 s
Epoch: 130       R²: -1.80       RMSE: 0.84      MAPE: 227.62    MAE: 0.55
Looptime: 0.063 s
Epoch: 135       R²: -1.80       RMSE: 0.84      MAPE: 227.79    MAE: 0.55
Looptime: 0.063 s
Epoch: 140       R²: -1.79       RMSE: 0.84      MAPE: 227.89    MAE: 0.55
Looptime: 0.063 s
Epoch: 145       R²: -1.78       RMSE: 0.84      MAPE: 228.13    MAE: 0.55
Looptime: 0.065 s
Epoch: 150       R²: -1.77       RMSE: 0.84      MAPE: 228.39    MAE: 0.55
Looptime: 0.065 s
Epoch: 155       R²: -1.77       RMSE: 0.84      MAPE: 228.70    MAE: 0.55
Looptime: 0.065 s
Epoch: 160       R²: -1.76       RMSE: 0.84      MAPE: 228.99    MAE: 0.55
Looptime: 0.063 s
Epoch: 165       R²: -1.75       RMSE: 0.84      MAPE: 229.16    MAE: 0.55
Looptime: 0.063 s
```

```
Epoch: 170      R²: -1.74       RMSE: 0.84      MAPE: 229.37    MAE: 0.55
Looptime: 0.063 s
Epoch: 175      R²: -1.74       RMSE: 0.84      MAPE: 229.52    MAE: 0.55
Looptime: 0.076 s
Epoch: 180      R²: -1.73       RMSE: 0.84      MAPE: 229.67    MAE: 0.55
Looptime: 0.076 s
Epoch: 185      R²: -1.73       RMSE: 0.84      MAPE: 229.84    MAE: 0.55
Looptime: 0.077 s
Epoch: 190      R²: -1.72       RMSE: 0.84      MAPE: 229.90    MAE: 0.55
Looptime: 0.076 s
Epoch: 195      R²: -1.71       RMSE: 0.84      MAPE: 230.04    MAE: 0.55
Looptime: 0.076 s
Epoch: 200      R²: -1.71       RMSE: 0.84      MAPE: 230.25    MAE: 0.55
Looptime: 0.076 s
Epoch: 205      R²: -1.70       RMSE: 0.84      MAPE: 230.37    MAE: 0.55
Looptime: 0.076 s
Epoch: 210      R²: -1.70       RMSE: 0.84      MAPE: 230.46    MAE: 0.55
Looptime: 0.076 s
Epoch: 215      R²: -1.69       RMSE: 0.84      MAPE: 230.55    MAE: 0.55
Looptime: 0.077 s
Epoch: 220      R²: -1.69       RMSE: 0.84      MAPE: 230.67    MAE: 0.55
Looptime: 0.064 s
Epoch: 225      R²: -1.68       RMSE: 0.84      MAPE: 230.68    MAE: 0.55
Looptime: 0.068 s
Epoch: 230      R²: -1.68       RMSE: 0.83      MAPE: 230.72    MAE: 0.55
Looptime: 0.068 s
Epoch: 235      R²: -1.67       RMSE: 0.83      MAPE: 230.96    MAE: 0.54
Looptime: 0.068 s
Epoch: 240      R²: -1.66       RMSE: 0.83      MAPE: 231.13    MAE: 0.54
Looptime: 0.067 s
Epoch: 245      R²: -1.66       RMSE: 0.83      MAPE: 231.26    MAE: 0.54
Looptime: 0.068 s
Epoch: 250      R²: -1.66       RMSE: 0.83      MAPE: 231.35    MAE: 0.54
Looptime: 0.068 s
Epoch: 255      R²: -1.65       RMSE: 0.83      MAPE: 231.47    MAE: 0.54
Looptime: 0.068 s
Epoch: 260      R²: -1.65       RMSE: 0.83      MAPE: 231.59    MAE: 0.54
Looptime: 0.067 s
Epoch: 265      R²: -1.64       RMSE: 0.83      MAPE: 231.68    MAE: 0.54
Looptime: 0.068 s
Epoch: 270      R²: -1.64       RMSE: 0.83      MAPE: 231.71    MAE: 0.54
Looptime: 0.068 s
Epoch: 275      R²: -1.63       RMSE: 0.83      MAPE: 231.92    MAE: 0.54
Looptime: 0.067 s
Epoch: 280      R²: -1.63       RMSE: 0.83      MAPE: 231.98    MAE: 0.54
Looptime: 0.073 s
Epoch: 285      R²: -1.62       RMSE: 0.83      MAPE: 232.21    MAE: 0.54
Looptime: 0.068 s
```

```
Epoch: 290       R²: -1.62       RMSE: 0.83      MAPE: 232.37    MAE: 0.54
Looptime: 0.068 s
Epoch: 295       R²: -1.61       RMSE: 0.83      MAPE: 232.53    MAE: 0.54
Looptime: 0.068 s
Epoch: 300       R²: -1.61       RMSE: 0.83      MAPE: 232.82    MAE: 0.54
Looptime: 0.068 s
Epoch: 305       R²: -1.60       RMSE: 0.83      MAPE: 232.98    MAE: 0.54
Looptime: 0.071 s
Epoch: 310       R²: -1.59       RMSE: 0.83      MAPE: 233.32    MAE: 0.54
Looptime: 0.068 s
Epoch: 315       R²: -1.59       RMSE: 0.83      MAPE: 233.54    MAE: 0.54
Looptime: 0.069 s
Epoch: 320       R²: -1.58       RMSE: 0.83      MAPE: 233.73    MAE: 0.54
Looptime: 0.068 s
Epoch: 325       R²: -1.58       RMSE: 0.83      MAPE: 233.88    MAE: 0.54
Looptime: 0.067 s
Epoch: 330       R²: -1.57       RMSE: 0.83      MAPE: 234.06    MAE: 0.54
Looptime: 0.067 s
Epoch: 335       R²: -1.57       RMSE: 0.82      MAPE: 234.13    MAE: 0.54
Looptime: 0.067 s
Epoch: 340       R²: -1.56       RMSE: 0.82      MAPE: 234.40    MAE: 0.54
Looptime: 0.067 s
Epoch: 345       R²: -1.56       RMSE: 0.82      MAPE: 234.59    MAE: 0.54
Looptime: 0.067 s
Epoch: 350       R²: -1.55       RMSE: 0.82      MAPE: 234.93    MAE: 0.54
Looptime: 0.068 s
Epoch: 355       R²: -1.55       RMSE: 0.82      MAPE: 235.08    MAE: 0.54
Looptime: 0.067 s
Epoch: 360       R²: -1.54       RMSE: 0.82      MAPE: 235.31    MAE: 0.54
Looptime: 0.068 s
Epoch: 365       R²: -1.54       RMSE: 0.82      MAPE: 235.45    MAE: 0.54
Looptime: 0.070 s
Epoch: 370       R²: -1.53       RMSE: 0.82      MAPE: 235.71    MAE: 0.54
Looptime: 0.070 s
Epoch: 375       R²: -1.53       RMSE: 0.82      MAPE: 235.90    MAE: 0.54
Looptime: 0.071 s
Epoch: 380       R²: -1.52       RMSE: 0.82      MAPE: 236.08    MAE: 0.54
Looptime: 0.070 s
Epoch: 385       R²: -1.52       RMSE: 0.82      MAPE: 236.31    MAE: 0.54
Looptime: 0.070 s
Epoch: 390       R²: -1.51       RMSE: 0.82      MAPE: 236.49    MAE: 0.54
Looptime: 0.072 s
Epoch: 395       R²: -1.50       RMSE: 0.82      MAPE: 236.78    MAE: 0.54
Looptime: 0.065 s
Epoch: 400       R²: -1.50       RMSE: 0.82      MAPE: 236.99    MAE: 0.53
Looptime: 0.064 s
Epoch: 405       R²: -1.49       RMSE: 0.82      MAPE: 237.28    MAE: 0.53
Looptime: 0.077 s
```

```
Epoch: 410       R²: -1.49       RMSE: 0.82      MAPE: 237.52    MAE: 0.53
Looptime: 0.063 s
Epoch: 415       R²: -1.48       RMSE: 0.82      MAPE: 237.67    MAE: 0.53
Looptime: 0.063 s
Epoch: 420       R²: -1.48       RMSE: 0.82      MAPE: 237.90    MAE: 0.53
Looptime: 0.063 s
Epoch: 425       R²: -1.47       RMSE: 0.82      MAPE: 238.09    MAE: 0.53
Looptime: 0.063 s
Epoch: 430       R²: -1.47       RMSE: 0.82      MAPE: 238.32    MAE: 0.53
Looptime: 0.063 s
Epoch: 435       R²: -1.47       RMSE: 0.82      MAPE: 238.62    MAE: 0.53
Looptime: 0.063 s
Epoch: 440       R²: -1.46       RMSE: 0.82      MAPE: 238.84    MAE: 0.53
Looptime: 0.064 s
Epoch: 445       R²: -1.46       RMSE: 0.82      MAPE: 239.02    MAE: 0.53
Looptime: 0.064 s
Epoch: 450       R²: -1.45       RMSE: 0.81      MAPE: 239.21    MAE: 0.53
Looptime: 0.064 s
Epoch: 455       R²: -1.45       RMSE: 0.81      MAPE: 239.37    MAE: 0.53
Looptime: 0.064 s
Epoch: 460       R²: -1.44       RMSE: 0.81      MAPE: 239.73    MAE: 0.53
Looptime: 0.064 s
Epoch: 465       R²: -1.44       RMSE: 0.81      MAPE: 239.92    MAE: 0.53
Looptime: 0.064 s
Epoch: 470       R²: -1.43       RMSE: 0.81      MAPE: 240.13    MAE: 0.53
Looptime: 0.064 s
Epoch: 475       R²: -1.43       RMSE: 0.81      MAPE: 240.39    MAE: 0.53
Looptime: 0.064 s
Epoch: 480       R²: -1.43       RMSE: 0.81      MAPE: 240.64    MAE: 0.53
Looptime: 0.066 s
Epoch: 485       R²: -1.42       RMSE: 0.81      MAPE: 240.90    MAE: 0.53
Looptime: 0.064 s
Epoch: 490       R²: -1.41       RMSE: 0.81      MAPE: 241.18    MAE: 0.53
Looptime: 0.064 s
Epoch: 495       R²: -1.41       RMSE: 0.81      MAPE: 241.55    MAE: 0.53
Looptime: 0.064 s
Epoch: 500       R²: -1.41       RMSE: 0.81      MAPE: 241.74    MAE: 0.53
Looptime: 0.064 s
Epoch: 505       R²: -1.40       RMSE: 0.81      MAPE: 242.07    MAE: 0.53
Looptime: 0.066 s
Epoch: 510       R²: -1.40       RMSE: 0.81      MAPE: 242.29    MAE: 0.53
Looptime: 0.064 s
Epoch: 515       R²: -1.39       RMSE: 0.81      MAPE: 242.50    MAE: 0.53
Looptime: 0.064 s
Epoch: 520       R²: -1.39       RMSE: 0.81      MAPE: 242.66    MAE: 0.53
Looptime: 0.063 s
Epoch: 525       R²: -1.39       RMSE: 0.81      MAPE: 242.79    MAE: 0.53
Looptime: 0.063 s
```

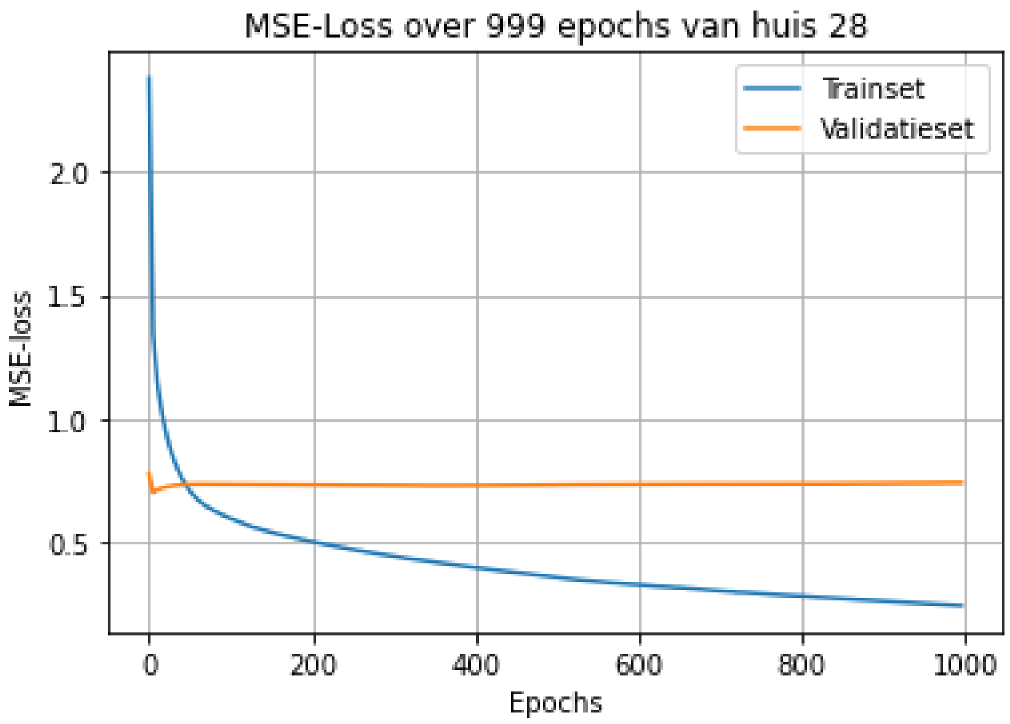
```
Epoch: 530       R²: -1.38       RMSE: 0.81      MAPE: 243.01    MAE: 0.53
Looptime: 0.063 s
Epoch: 535       R²: -1.37       RMSE: 0.81      MAPE: 243.35    MAE: 0.53
Looptime: 0.063 s
Epoch: 540       R²: -1.37       RMSE: 0.81      MAPE: 243.51    MAE: 0.53
Looptime: 0.063 s
Epoch: 545       R²: -1.37       RMSE: 0.81      MAPE: 243.72    MAE: 0.53
Looptime: 0.063 s
Epoch: 550       R²: -1.36       RMSE: 0.81      MAPE: 243.84    MAE: 0.53
Looptime: 0.071 s
Epoch: 555       R²: -1.36       RMSE: 0.81      MAPE: 243.96    MAE: 0.53
Looptime: 0.073 s
Epoch: 560       R²: -1.36       RMSE: 0.81      MAPE: 244.18    MAE: 0.53
Looptime: 0.073 s
Epoch: 565       R²: -1.35       RMSE: 0.81      MAPE: 244.28    MAE: 0.53
Looptime: 0.073 s
Epoch: 570       R²: -1.35       RMSE: 0.81      MAPE: 244.46    MAE: 0.53
Looptime: 0.073 s
Epoch: 575       R²: -1.34       RMSE: 0.81      MAPE: 244.75    MAE: 0.53
Looptime: 0.073 s
Epoch: 580       R²: -1.34       RMSE: 0.80      MAPE: 244.91    MAE: 0.53
Looptime: 0.063 s
Epoch: 585       R²: -1.33       RMSE: 0.80      MAPE: 245.16    MAE: 0.53
Looptime: 0.063 s
Epoch: 590       R²: -1.33       RMSE: 0.80      MAPE: 245.37    MAE: 0.52
Looptime: 0.063 s
Epoch: 595       R²: -1.33       RMSE: 0.80      MAPE: 245.52    MAE: 0.52
Looptime: 0.065 s
Epoch: 600       R²: -1.32       RMSE: 0.80      MAPE: 245.74    MAE: 0.52
Looptime: 0.063 s
Epoch: 605       R²: -1.32       RMSE: 0.80      MAPE: 245.88    MAE: 0.52
Looptime: 0.063 s
Epoch: 610       R²: -1.31       RMSE: 0.80      MAPE: 246.11    MAE: 0.52
Looptime: 0.063 s
Epoch: 615       R²: -1.31       RMSE: 0.80      MAPE: 246.32    MAE: 0.52
Looptime: 0.063 s
Epoch: 620       R²: -1.30       RMSE: 0.80      MAPE: 246.50    MAE: 0.52
Looptime: 0.063 s
Epoch: 625       R²: -1.30       RMSE: 0.80      MAPE: 246.86    MAE: 0.52
Looptime: 0.063 s
Epoch: 630       R²: -1.30       RMSE: 0.80      MAPE: 247.05    MAE: 0.52
Looptime: 0.063 s
Epoch: 635       R²: -1.29       RMSE: 0.80      MAPE: 247.39    MAE: 0.52
Looptime: 0.062 s
Epoch: 640       R²: -1.29       RMSE: 0.80      MAPE: 247.65    MAE: 0.52
Looptime: 0.063 s
Epoch: 645       R²: -1.28       RMSE: 0.80      MAPE: 247.94    MAE: 0.52
Looptime: 0.063 s
```

```
Epoch: 650       R²: -1.28       RMSE: 0.80      MAPE: 248.15    MAE: 0.52
Looptime: 0.075 s
Epoch: 655       R²: -1.27       RMSE: 0.80      MAPE: 248.31    MAE: 0.52
Looptime: 0.063 s
Epoch: 660       R²: -1.27       RMSE: 0.80      MAPE: 248.40    MAE: 0.52
Looptime: 0.063 s
Epoch: 665       R²: -1.27       RMSE: 0.80      MAPE: 248.62    MAE: 0.52
Looptime: 0.063 s
Epoch: 670       R²: -1.26       RMSE: 0.80      MAPE: 248.85    MAE: 0.52
Looptime: 0.063 s
Epoch: 675       R²: -1.26       RMSE: 0.80      MAPE: 249.03    MAE: 0.52
Looptime: 0.063 s
Epoch: 680       R²: -1.25       RMSE: 0.80      MAPE: 249.17    MAE: 0.52
Looptime: 0.063 s
Epoch: 685       R²: -1.25       RMSE: 0.80      MAPE: 249.33    MAE: 0.52
Looptime: 0.063 s
Epoch: 690       R²: -1.24       RMSE: 0.80      MAPE: 249.45    MAE: 0.52
Looptime: 0.063 s
Epoch: 695       R²: -1.24       RMSE: 0.80      MAPE: 249.48    MAE: 0.52
Looptime: 0.063 s
Epoch: 700       R²: -1.23       RMSE: 0.80      MAPE: 249.69    MAE: 0.52
Looptime: 0.063 s
Epoch: 705       R²: -1.23       RMSE: 0.80      MAPE: 249.95    MAE: 0.52
Looptime: 0.063 s
Epoch: 710       R²: -1.22       RMSE: 0.80      MAPE: 250.19    MAE: 0.52
Looptime: 0.063 s
Epoch: 715       R²: -1.22       RMSE: 0.79      MAPE: 250.42    MAE: 0.52
Looptime: 0.063 s
Epoch: 720       R²: -1.21       RMSE: 0.79      MAPE: 250.60    MAE: 0.52
Looptime: 0.063 s
Epoch: 725       R²: -1.21       RMSE: 0.79      MAPE: 250.80    MAE: 0.52
Looptime: 0.063 s
Epoch: 730       R²: -1.21       RMSE: 0.79      MAPE: 251.07    MAE: 0.52
Looptime: 0.063 s
Epoch: 735       R²: -1.20       RMSE: 0.79      MAPE: 251.25    MAE: 0.52
Looptime: 0.063 s
Epoch: 740       R²: -1.20       RMSE: 0.79      MAPE: 251.46    MAE: 0.52
Looptime: 0.063 s
Epoch: 745       R²: -1.20       RMSE: 0.79      MAPE: 251.52    MAE: 0.52
Looptime: 0.063 s
Epoch: 750       R²: -1.19       RMSE: 0.79      MAPE: 251.80    MAE: 0.52
Looptime: 0.063 s
Epoch: 755       R²: -1.18       RMSE: 0.79      MAPE: 251.99    MAE: 0.52
Looptime: 0.065 s
Epoch: 760       R²: -1.18       RMSE: 0.79      MAPE: 252.21    MAE: 0.52
Looptime: 0.067 s
Epoch: 765       R²: -1.18       RMSE: 0.79      MAPE: 252.50    MAE: 0.52
Looptime: 0.064 s
```

```
Epoch: 770       R²: -1.18       RMSE: 0.79      MAPE: 252.82    MAE: 0.52
Looptime: 0.064 s
Epoch: 775       R²: -1.17       RMSE: 0.79      MAPE: 253.03    MAE: 0.52
Looptime: 0.064 s
Epoch: 780       R²: -1.17       RMSE: 0.79      MAPE: 253.26    MAE: 0.52
Looptime: 0.064 s
Epoch: 785       R²: -1.16       RMSE: 0.79      MAPE: 253.57    MAE: 0.52
Looptime: 0.064 s
Epoch: 790       R²: -1.16       RMSE: 0.79      MAPE: 253.65    MAE: 0.52
Looptime: 0.064 s
Epoch: 795       R²: -1.15       RMSE: 0.79      MAPE: 253.97    MAE: 0.52
Looptime: 0.064 s
Epoch: 800       R²: -1.15       RMSE: 0.79      MAPE: 254.19    MAE: 0.52
Looptime: 0.064 s
Epoch: 805       R²: -1.15       RMSE: 0.79      MAPE: 254.44    MAE: 0.52
Looptime: 0.062 s
Epoch: 810       R²: -1.14       RMSE: 0.79      MAPE: 254.72    MAE: 0.52
Looptime: 0.064 s
Epoch: 815       R²: -1.14       RMSE: 0.79      MAPE: 254.80    MAE: 0.51
Looptime: 0.062 s
Epoch: 820       R²: -1.13       RMSE: 0.79      MAPE: 254.94    MAE: 0.51
Looptime: 0.062 s
Epoch: 825       R²: -1.13       RMSE: 0.79      MAPE: 255.07    MAE: 0.51
Looptime: 0.067 s
Epoch: 830       R²: -1.12       RMSE: 0.79      MAPE: 255.29    MAE: 0.51
Looptime: 0.061 s
Epoch: 835       R²: -1.12       RMSE: 0.79      MAPE: 255.35    MAE: 0.51
Looptime: 0.061 s
Epoch: 840       R²: -1.12       RMSE: 0.79      MAPE: 255.41    MAE: 0.51
Looptime: 0.075 s
Epoch: 845       R²: -1.11       RMSE: 0.79      MAPE: 255.62    MAE: 0.51
Looptime: 0.065 s
Epoch: 850       R²: -1.11       RMSE: 0.79      MAPE: 255.88    MAE: 0.51
Looptime: 0.065 s
Epoch: 855       R²: -1.11       RMSE: 0.79      MAPE: 255.91    MAE: 0.51
Looptime: 0.065 s
Epoch: 860       R²: -1.10       RMSE: 0.79      MAPE: 255.98    MAE: 0.51
Looptime: 0.065 s
Epoch: 865       R²: -1.10       RMSE: 0.79      MAPE: 256.08    MAE: 0.51
Looptime: 0.065 s
Epoch: 870       R²: -1.09       RMSE: 0.78      MAPE: 256.18    MAE: 0.51
Looptime: 0.065 s
Epoch: 875       R²: -1.09       RMSE: 0.78      MAPE: 256.21    MAE: 0.51
Looptime: 0.065 s
Epoch: 880       R²: -1.09       RMSE: 0.78      MAPE: 256.51    MAE: 0.51
Looptime: 0.065 s
Epoch: 885       R²: -1.08       RMSE: 0.78      MAPE: 256.71    MAE: 0.51
Looptime: 0.063 s
```

```
Epoch: 890       R²: -1.08       RMSE: 0.78      MAPE: 256.71    MAE: 0.51
Looptime: 0.064 s
Epoch: 895       R²: -1.08       RMSE: 0.78      MAPE: 256.83    MAE: 0.51
Looptime: 0.064 s
Epoch: 900       R²: -1.07       RMSE: 0.78      MAPE: 256.94    MAE: 0.51
Looptime: 0.064 s
Epoch: 905       R²: -1.07       RMSE: 0.78      MAPE: 257.22    MAE: 0.51
Looptime: 0.064 s
Epoch: 910       R²: -1.06       RMSE: 0.78      MAPE: 257.37    MAE: 0.51
Looptime: 0.067 s
Epoch: 915       R²: -1.06       RMSE: 0.78      MAPE: 257.56    MAE: 0.51
Looptime: 0.064 s
Epoch: 920       R²: -1.06       RMSE: 0.78      MAPE: 257.73    MAE: 0.51
Looptime: 0.064 s
Epoch: 925       R²: -1.05       RMSE: 0.78      MAPE: 257.90    MAE: 0.51
Looptime: 0.065 s
Epoch: 930       R²: -1.05       RMSE: 0.78      MAPE: 258.26    MAE: 0.51
Looptime: 0.064 s
Epoch: 935       R²: -1.05       RMSE: 0.78      MAPE: 258.31    MAE: 0.51
Looptime: 0.065 s
Epoch: 940       R²: -1.04       RMSE: 0.78      MAPE: 258.46    MAE: 0.51
Looptime: 0.064 s
Epoch: 945       R²: -1.04       RMSE: 0.78      MAPE: 258.68    MAE: 0.51
Looptime: 0.063 s
Epoch: 950       R²: -1.03       RMSE: 0.78      MAPE: 258.83    MAE: 0.51
Looptime: 0.064 s
Epoch: 955       R²: -1.03       RMSE: 0.78      MAPE: 258.96    MAE: 0.51
Looptime: 0.064 s
Epoch: 960       R²: -1.03       RMSE: 0.78      MAPE: 259.04    MAE: 0.51
Looptime: 0.064 s
Epoch: 965       R²: -1.02       RMSE: 0.78      MAPE: 259.20    MAE: 0.51
Looptime: 0.064 s
Epoch: 970       R²: -1.02       RMSE: 0.78      MAPE: 259.27    MAE: 0.51
Looptime: 0.064 s
Epoch: 975       R²: -1.02       RMSE: 0.78      MAPE: 259.56    MAE: 0.51
Looptime: 0.065 s
Epoch: 980       R²: -1.02       RMSE: 0.78      MAPE: 259.57    MAE: 0.51
Looptime: 0.065 s
Epoch: 985       R²: -1.01       RMSE: 0.78      MAPE: 259.86    MAE: 0.51
Looptime: 0.065 s
Epoch: 990       R²: -1.01       RMSE: 0.78      MAPE: 260.05    MAE: 0.51
Looptime: 0.065 s
Epoch: 995       R²: -1.00       RMSE: 0.78      MAPE: 260.25    MAE: 0.51
Looptime: 0.070 s
```

[10]:
```
%matplotlib inline
plt.plot([i*show_every for i in range(0,len(alijst))],alijst,label="Trainset")
plt.plot([i*show_every for i in range(0,len(blijst))],blijst,␣
↪label="Validatieset")

plt.title('MSE-Loss over ' +str(list(epochs)[-1])+ ' epochs van huis 28' )
plt.xlabel("Epochs")
plt.ylabel("MSE-loss") # MSELoss
plt.legend()
#plt.xlim([150,250])
#plt.ylim([0.5,0.8])
plt.grid()
plt.show()
```



1.7 Test het Neuraal Netwerk:

Geïmplementeerde validatie methoden: - R^2 - RMSE - MAPE - MAE (L1-Loss)

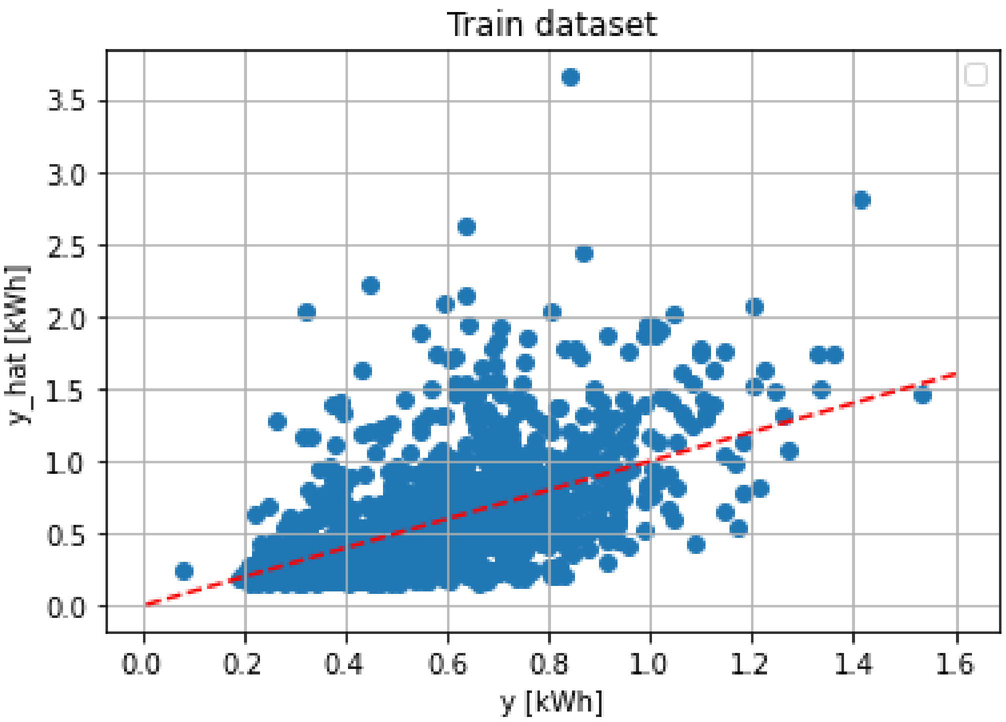
Train evaluation

[11]:
```
model.eval()

y = scalery.inverse_transform(train_y_t.cpu().detach().numpy())
yhat = scalery.inverse_transform(model(train_X_t.float()).cpu().detach().
↪numpy())
```

```
# R^2-score:
r2 = r2_score(yhat, y)
# RMSE:
rmse = np.sqrt(mean_squared_error(yhat, y))
# MAPE:
actual, pred = np.array(y), np.array(yhat)
mape = np.mean(np.abs((actual - pred) / actual)) * 100
# MAE:
mae = mean_absolute_error(yhat, y)


#zet de voorspelde tegen de werkelijke waarde uit.
%matplotlib inline
plt.scatter(yhat,y)
plt.xlabel("y [kWh]")
plt.ylabel("y_hat [kWh]")
plt.legend(loc="upper right")
plt.title('Train dataset')
plt.plot(plt.xlim(), plt.xlim(), ls="--", c='r', label="$y$=$\hat{y}$")
# plt.xlim([0.1,1])
# plt.ylim([0.1,1])
print('R\u00b2: \t%.2f \nRMSE: \t%.2f \nMAPE: \t%.2f \nMAE: \t%.2f' %␣
 ↪(r2,rmse,mape,mae))
plt.grid()
plt.show()
```

```
No handles with labels found to put in legend.
```

```
R²:     -1.00
RMSE:   0.29
MAPE:   40.60
MAE:    0.19
```

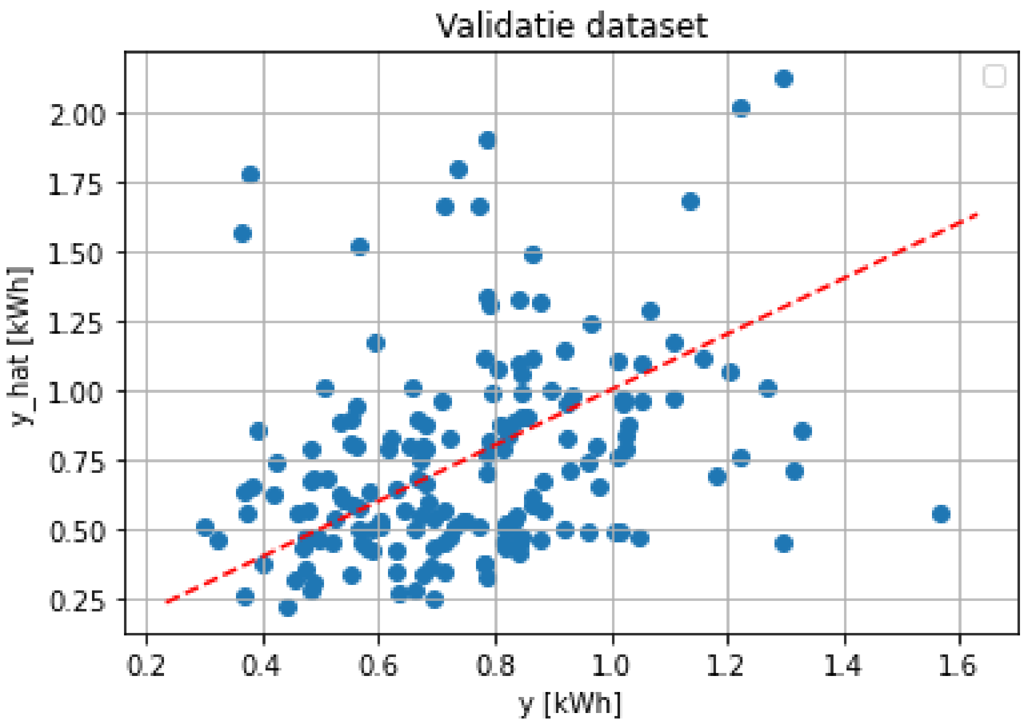


Validation evaluation

```
[12]: model.eval()

y = scalery.inverse_transform(valid_y_t.cpu().detach().numpy())
yhat = scalery.inverse_transform(model(valid_X_t.float()).cpu().detach().
 ↪numpy())

# R^2-score:
r2 = r2_score(yhat, y)
# RMSE:
rmse = np.sqrt(mean_squared_error(yhat, y))
# MAPE:
actual, pred = np.array(y), np.array(yhat)
mape = np.mean(np.abs((actual - pred) / actual)) * 100
# MAE:
mae = mean_absolute_error(yhat, y)

#zet de voorspelde tegen de werkelijke waarde uit.
%matplotlib inline
plt.scatter(yhat,y)
plt.xlabel("y [kWh]")
plt.ylabel("y_hat [kWh]")
plt.legend(loc="upper right")
plt.title("Validatie dataset")
plt.plot(plt.xlim(), plt.xlim(), ls="--", c='r', label="$y$=$\hat{y}$")
# plt.xlim([0.1,1])
# plt.ylim([0.1,1])
print('R\u00b2: \t%.2f \nRMSE: \t%.2f \nMAPE: \t%.2f \nMAE: \t%.2f' %␣
 ↪(r2,rmse,mape,mae))
plt.grid()
plt.show()
```

```
No handles with labels found to put in legend.
```

```
R²:     -1.48
RMSE:   0.37
MAPE:   41.22
MAE:    0.27
```



```
[13]: A = np.sum(scalery.inverse_transform(model(train_X_t.float()).cpu().detach().
       ↪numpy()))
      B = np.sum(scalery.inverse_transform(train_y_t.cpu().detach().numpy()))
      C = (A-B)/B*100

      D = np.sum(scalery.inverse_transform(model(valid_X_t.float()).cpu().detach().
       ↪numpy()))
```

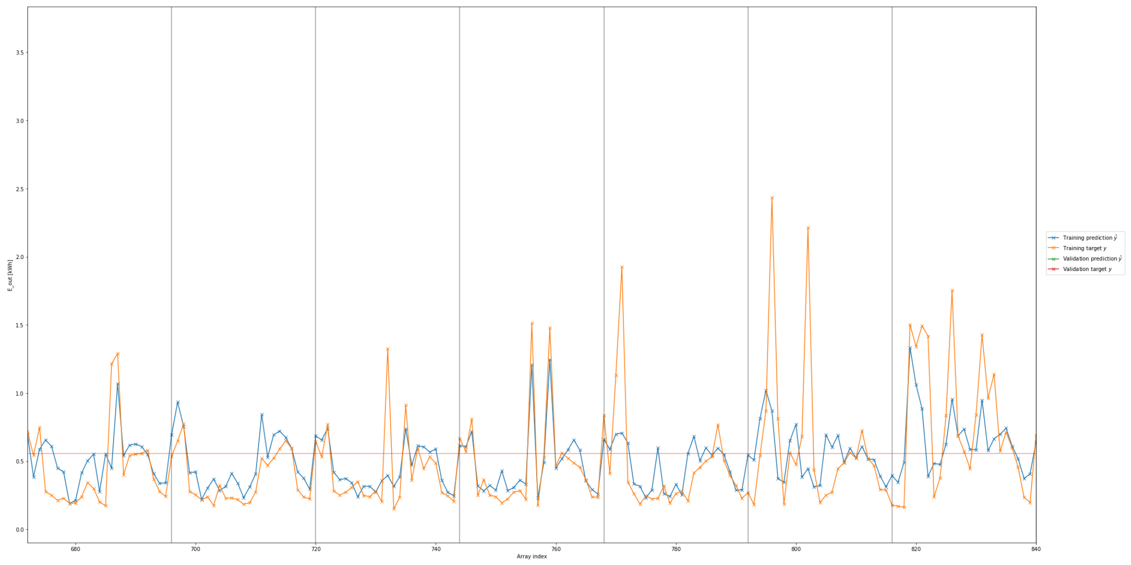
```
E = np.sum(scalery.inverse_transform(valid_y_t.cpu().detach().numpy()))
F = (D-E)/E*100

print("Train:\t\t Model = %.2f kWh \t Actual = %.2f kWh \t percentage = %.2f" ␣
↪% (A,B,C) +"%")
print("Validation:\t Model = %.2f kWh \t Actual = %.2f kWh \t percentage = %.
↪2f" % (D,E,F) +"%")
```

```
Train:          Model = 1014.46 kWh     Actual = 1001.73 kWh    percentage =
1.27%
Validation:     Model = 126.34 kWh      Actual = 126.92 kWh     percentage =
-0.46%
```

[14]:
```
kaas = []
#kaas.append(([i[0] for i in train_y_t.detach().cpu().numpy().tolist()])[-1:
↪][0])
for i in [i[0] for i in scalery.inverse_transform((model(valid_X_t.float()).
↪detach().cpu().numpy()).tolist())]:
    kaas.append(i)

kaas1 = []
#kaas1.append(((train_y_t.detach().cpu().numpy().tolist())[-1:][0][0]))
for i in [i[0] for i in scalery.inverse_transform((valid_y_t.detach().cpu().
↪numpy()).tolist())]:
    kaas1.append(i)
```

[15]:
```
# Training
%matplotlib inline
plt.subplots(figsize=(30,15))
plt.plot(np.arange(0,len(train_y_t.detach().cpu().numpy())), scalery.
↪inverse_transform(model(train_X_t.float()).detach().cpu().numpy()),
         "x-", label="Training prediction $\hat{y}$")
plt.plot(np.arange(0,len(train_y_t.detach().cpu().numpy())), scalery.
↪inverse_transform(train_y_t.detach().cpu().numpy()),
         "x-", label="Training target $y$")
plt.grid()
plt.ylabel("E_out [kWh]")
plt.legend(loc=(1.01, 0.5))
plt.plot((np.arange(len(train_y_t.detach().cpu().numpy()),len(train_y_t.
↪detach().cpu().numpy())+len(valid_y_t.detach().cpu().numpy()))).tolist(),␣
↪kaas,
         "x-", label="Validation prediction $\hat{y}$")
plt.plot((np.arange(len(train_y_t.detach().cpu().numpy()),len(train_y_t.
↪detach().cpu().numpy())+len(valid_y_t.detach().cpu().numpy()))).tolist(),␣
↪kaas1,
         "x-", label="Validation target $y$")
```

```
[plt.axvline(i,color="black", alpha=0.4) for i in list(range(0,len(valid_y_t.
 ↪detach().cpu().numpy())+len(train_y_t.detach().cpu().numpy())+24,24))]
plt.axhline(np.mean(scalery.inverse_transform(train_y_t.detach().cpu().numpy().
 ↪tolist())),color='red', alpha=0.4)
# layout
# plt.ylim([-1,1.5])
plt.xlim([672,840])
# plt.xlim([336,672])
plt.xlabel("Array index")
plt.ylabel("E_out [kWh]")
plt.legend(loc=(1.01, 0.5))
plt.grid()
plt.tight_layout()
plt.show()
```



2 MVLR

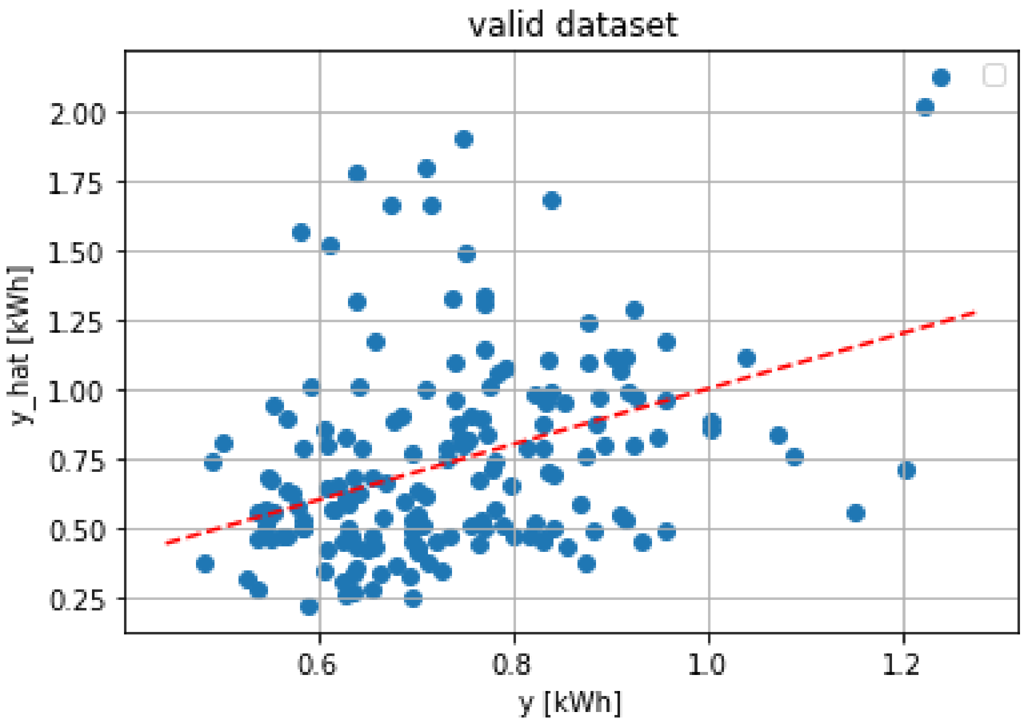
[16]:
```
from sklearn import linear_model
regr = linear_model.LinearRegression()
regr.fit(train_X,train_y)
yhat =  scalery.inverse_transform(regr.predict(valid_X))
y = scalery.inverse_transform(valid_y)
```

[17]:
```
r2 = r2_score(yhat, y)
rmse = np.sqrt(mean_squared_error(yhat, y))
actual, pred = np.array(y), np.array(yhat)
```

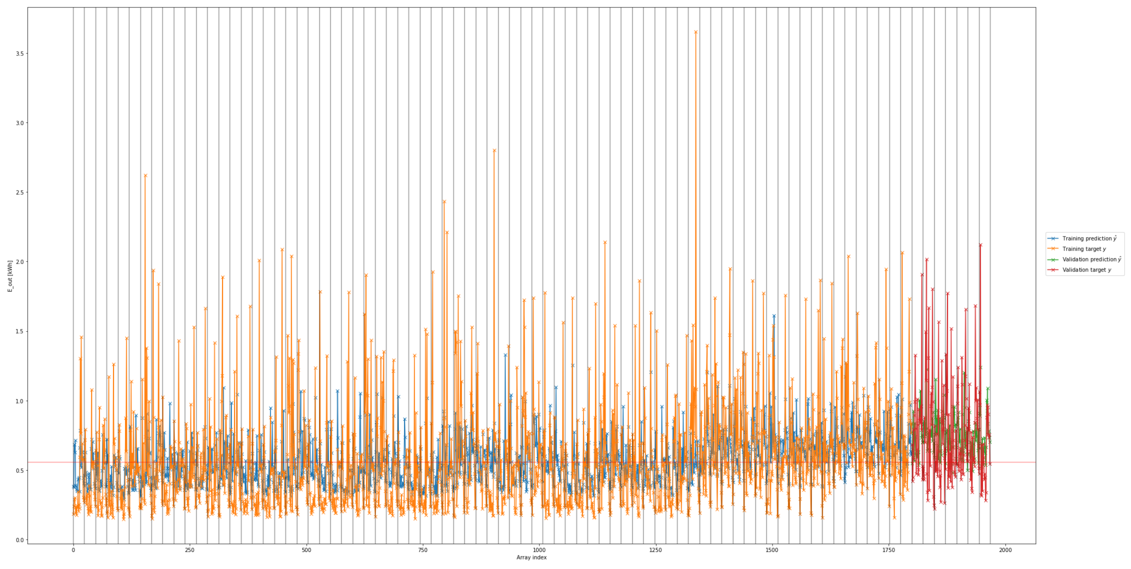
```
mape = np.mean(np.abs((actual - pred) / actual)) * 100
mae = mean_absolute_error(yhat, y)

#zet de voorspelde tegen de werkelijke waarde uit.
%matplotlib inline
plt.scatter(yhat,y)
plt.xlabel("y [kWh]")
plt.ylabel("y_hat [kWh]")
plt.legend(loc="upper right")
plt.title('valid dataset')
plt.plot(plt.xlim(), plt.xlim(), ls="--", c='r', label="$y$=$\hat{y}$")
# plt.xlim([0.1,1])
# plt.ylim([0.1,1])
print('R\u00b2: \t%.2f \nRMSE: \t%.2f \nMAPE: \t%.2f \nMAE: \t%.2f' %␣
 ↪(r2,rmse,mape,mae))
plt.grid()
plt.show()
```

```
No handles with labels found to put in legend.
```

```
R²:     -4.62
RMSE:   0.35
MAPE:   38.90
MAE:    0.26
```



[18]:
```
# Training
%matplotlib inline
plt.subplots(figsize=(30,15))
plt.plot(np.arange(0,len(train_y_t.detach().cpu().numpy())), scalery.
 ↪inverse_transform(regr.predict(train_X)),
         "x-", label="Training prediction $\hat{y}$")
plt.plot(np.arange(0,len(train_y_t.detach().cpu().numpy())), scalery.
 ↪inverse_transform(train_y_t.detach().cpu().numpy()),
         "x-", label="Training target $y$")
plt.grid()
plt.ylabel("E_out [kWh]")
plt.legend(loc=(1.01, 0.5))
plt.plot((np.arange(len(train_y_t.detach().cpu().numpy()),len(train_y_t.
 ↪detach().cpu().numpy())+len(valid_y_t.detach().cpu().numpy()))).tolist(),␣
 ↪scalery.inverse_transform(regr.predict(valid_X)),
         "x-", label="Validation prediction $\hat{y}$")
plt.plot((np.arange(len(train_y_t.detach().cpu().numpy()),len(train_y_t.
 ↪detach().cpu().numpy())+len(valid_y_t.detach().cpu().numpy()))).tolist(),␣
 ↪scalery.inverse_transform(valid_y),
         "x-", label="Validation target $y$")
[plt.axvline(i,color="black", alpha=0.4) for i in list(range(0,len(valid_y_t.
 ↪detach().cpu().numpy())+len(train_y_t.detach().cpu().numpy())+24,24))]
plt.axhline(np.mean(scalery.inverse_transform(train_y_t.detach().cpu().numpy().
 ↪tolist())),color='red', alpha=0.4)
# layout
# plt.ylim([-1,1.5])
#plt.xlim([672,840])
# plt.xlim([336,672])
plt.xlabel("Array index")
plt.ylabel("E_out [kWh]")
plt.legend(loc=(1.01, 0.5))
plt.grid()
plt.tight_layout()
plt.show()
```



3 SVR

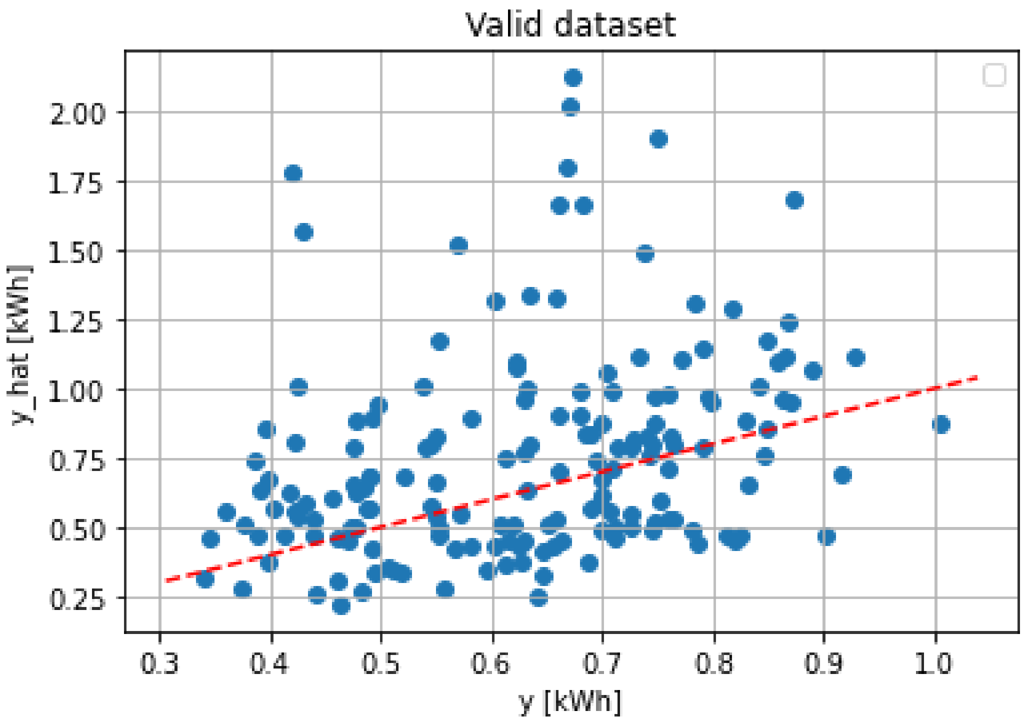
[19]:
```
from sklearn.svm import SVR
regr = SVR()
regr.fit(train_X,train_y)
yhat =  scalery.inverse_transform(regr.predict(valid_X))
y = scalery.inverse_transform(valid_y)
```

```
/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)
```

[20]:
```
r2 = r2_score(yhat, y)
rmse = np.sqrt(mean_squared_error(yhat, y))
actual, pred = np.array(y), np.array(yhat)
mape = np.mean(np.abs((actual - pred) / actual)) * 100
mae = mean_absolute_error(yhat, y)

#zet de voorspelde tegen de werkelijke waarde uit.
%matplotlib inline
plt.scatter(yhat,y)
plt.xlabel("y [kWh]")
plt.ylabel("y_hat [kWh]")
plt.legend(loc="upper right")
plt.title('Valid dataset')
plt.plot(plt.xlim(), plt.xlim(), ls="--", c='r', label="$y$=$\hat{y}$")
# plt.xlim([0.1,1])
# plt.ylim([0.1,1])
print('R\u00b2: \t%.2f \nRMSE: \t%.2f \nMAPE: \t%.2f \nMAE: \t%.2f' %␣
 ↪(r2,rmse,mape,mae))
plt.grid()
plt.show()
```

```
No handles with labels found to put in legend.
```

```
R²:     -5.46
RMSE:   0.37
MAPE:   41.35
MAE:    0.26
```

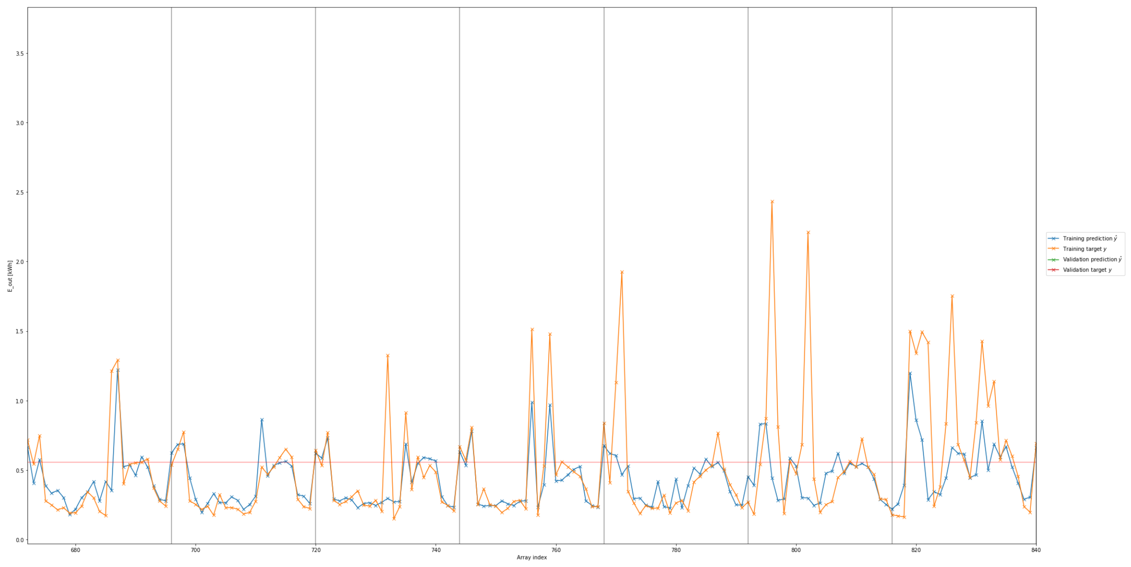


[21]:
```
A = np.sum(scalery.inverse_transform(regr.predict(train_X)))
B = np.sum(scalery.inverse_transform(train_y))
C = (A-B)/B*100

D = np.sum(scalery.inverse_transform(regr.predict(valid_X)))
E = np.sum(scalery.inverse_transform(valid_y))
F = (D-E)/E*100
```

```
print("Train:\t\t Model = %.2f kWh \t Actual = %.2f kWh \t percentage = %.2f" ␣
↪% (A,B,C) +"%")
print("Validation:\t Model = %.2f kWh \t Actual = %.2f kWh \t percentage = %.
↪2f" % (D,E,F) +"%")
```

```
Train:          Model = 880.63 kWh      Actual = 1001.73 kWh     percentage =
-12.09%
Validation:     Model = 105.74 kWh      Actual = 126.92 kWh      percentage =
-16.69%
```

[22]:
```
# Training
%matplotlib inline
plt.subplots(figsize=(30,15))
plt.plot(np.arange(0,len(train_y_t.detach().cpu().numpy())), scalery.
↪inverse_transform(regr.predict(train_X)),
         "x-", label="Training prediction $\hat{y}$")
plt.plot(np.arange(0,len(train_y_t.detach().cpu().numpy())), scalery.
↪inverse_transform(train_y_t.detach().cpu().numpy()),
         "x-", label="Training target $y$")
plt.grid()
plt.ylabel("E_out [kWh]")
plt.legend(loc=(1.01, 0.5))
plt.plot((np.arange(len(train_y_t.detach().cpu().numpy()),len(train_y_t.
↪detach().cpu().numpy())+len(valid_y_t.detach().cpu().numpy()))).tolist(),␣
↪scalery.inverse_transform(regr.predict(valid_X)),
         "x-", label="Validation prediction $\hat{y}$")
plt.plot((np.arange(len(train_y_t.detach().cpu().numpy()),len(train_y_t.
↪detach().cpu().numpy())+len(valid_y_t.detach().cpu().numpy()))).tolist(),␣
↪scalery.inverse_transform(valid_y),
         "x-", label="Validation target $y$")
[plt.axvline(i,color="black", alpha=0.4) for i in list(range(0,len(valid_y_t.
↪detach().cpu().numpy())+len(train_y_t.detach().cpu().numpy())+24,24))]
plt.axhline(np.mean(scalery.inverse_transform(train_y_t.detach().cpu().numpy().
↪tolist())),color='red', alpha=0.4)
# layout
#plt.ylim([-1,1.5])
plt.xlim([672,840])
#plt.xlim([336,672])
plt.xlabel("Array index")
plt.ylabel("E_out [kWh]")
plt.legend(loc=(1.01, 0.5))
plt.grid()
plt.tight_layout()
plt.show()
```



3.1 Stop de notebook:

//%%javascript //Jupyter.notebook.session.delete()