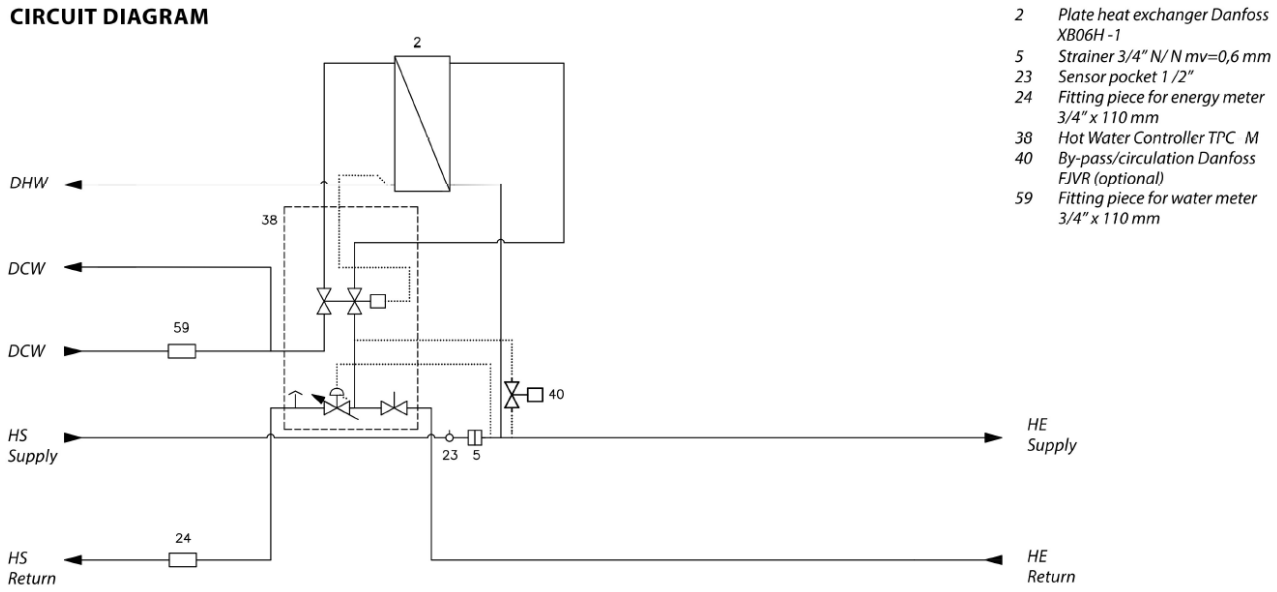
Supplementary Information

Heat Demand Forecast for Residential Units with different Online and Offline Machine Learning Methods

Section 1: Hydraulic diagram of the dwelling station.



Hydraulic diagram of a dwelling station. The figure shows a Danfoss Evoflat MSS DE3 TPC-M[1] station. In the Puls-G building , Stiebel Eltron WSG-2-DUO[2] stations are installed with a similar design. The used meter is located at the marked point 24.

[1] https://store.danfoss.com/en/Climate-Solutions-for-heating/Stations/Direct-Heating-with-Mixing-and-Domestic-Hot-Water-EvoFlat-MSS%2C-Type-3%2C-10-bar%2C-95-%C2%B0C%2C-DHW-controller-name%3A-TPC-M%2C-Thermostat/p/145B1722

[2] https://www.stiebel-eltron.de/content/dam/ste/cdbassets/historic/bedienungs-_u_installationsanleitungen/WSP_WSG-2-DUO__94d875e8-8076-42c1-9c4d-d725af656644.pdf

Section 2: Figure of the input parameter

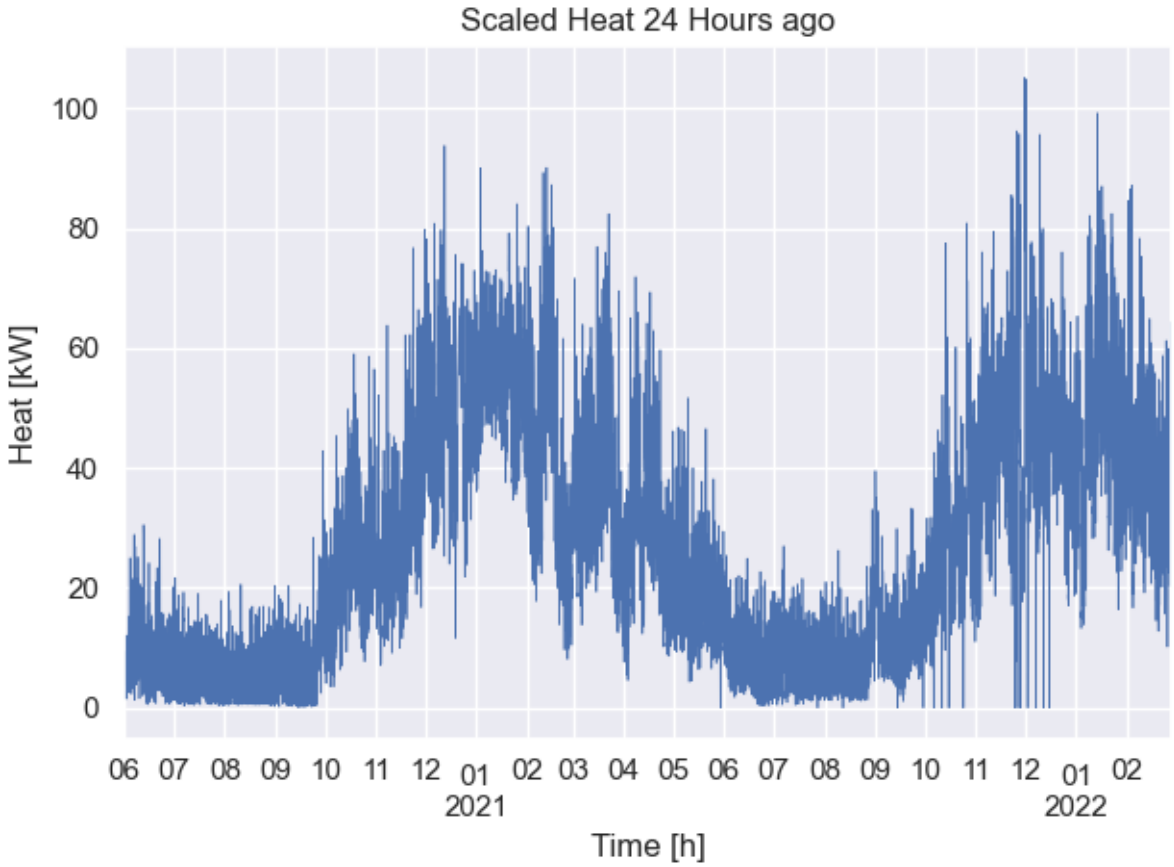


Figure 2.1: The hourly scaled heat consumption 24 hours ago measured in kilowatts [kW] from 1. June 2020, to 28. February 2022. Missing values are filled in with 0.

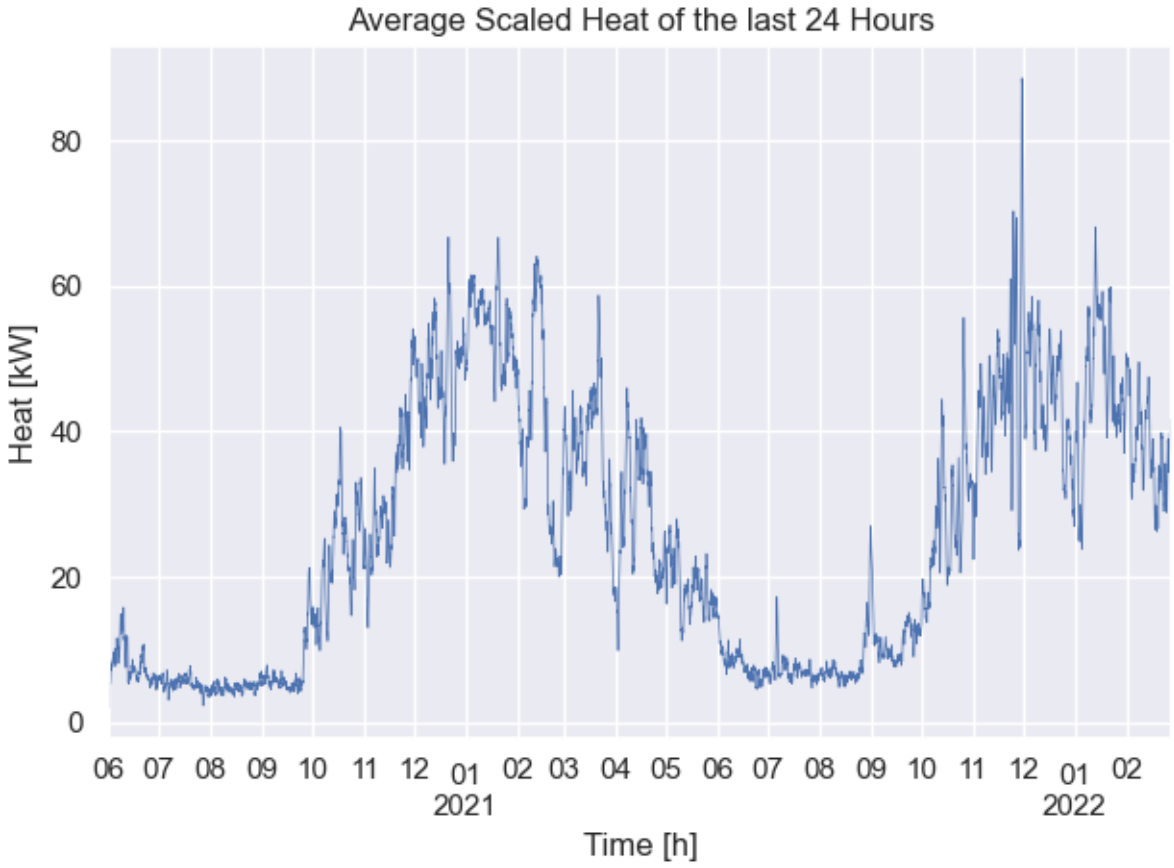


Figure 2.2: The hourly average scaled heat consumption of 24 hours measured in kilowatts [kW] from 1. June 2020, to 28. February 2022.

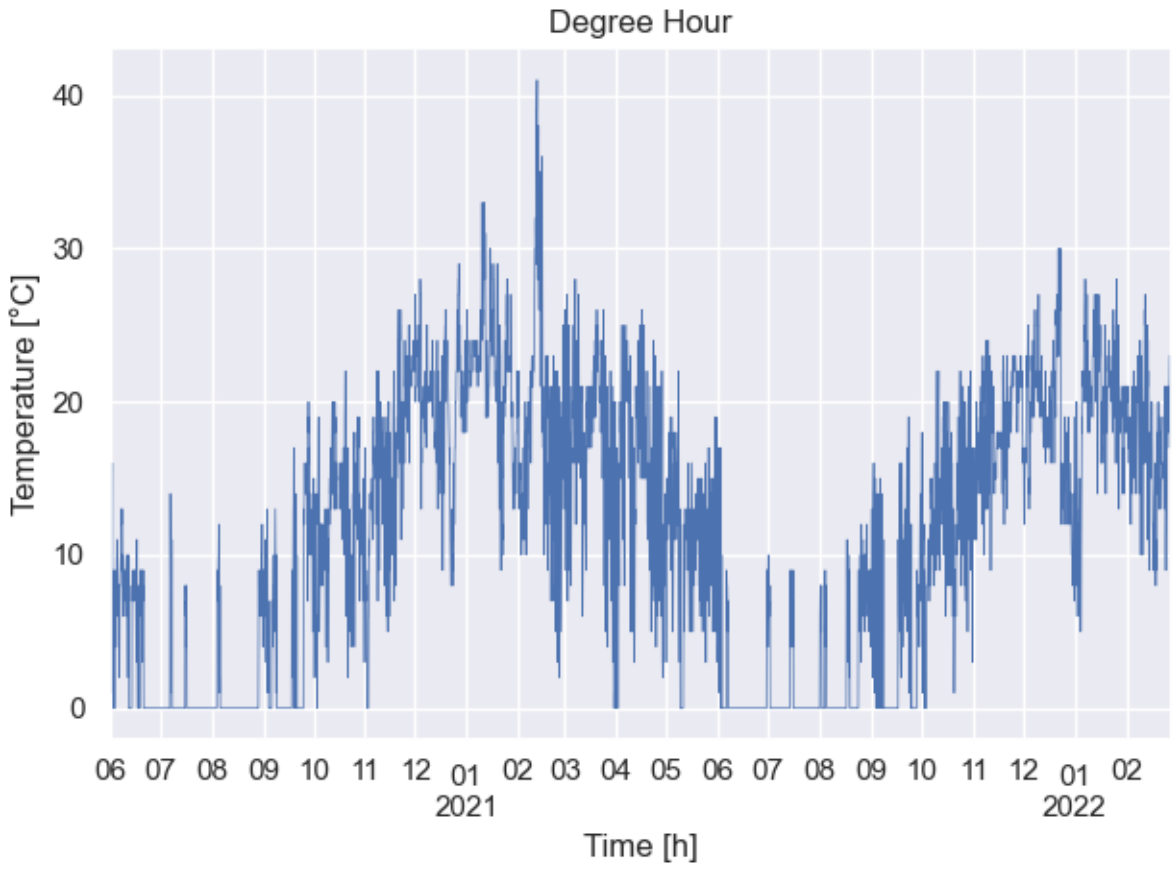


Figure 2.3: The hourly degree hour measured in degrees Celsius [°C] from 1. June 2020, to 28. February 2022.

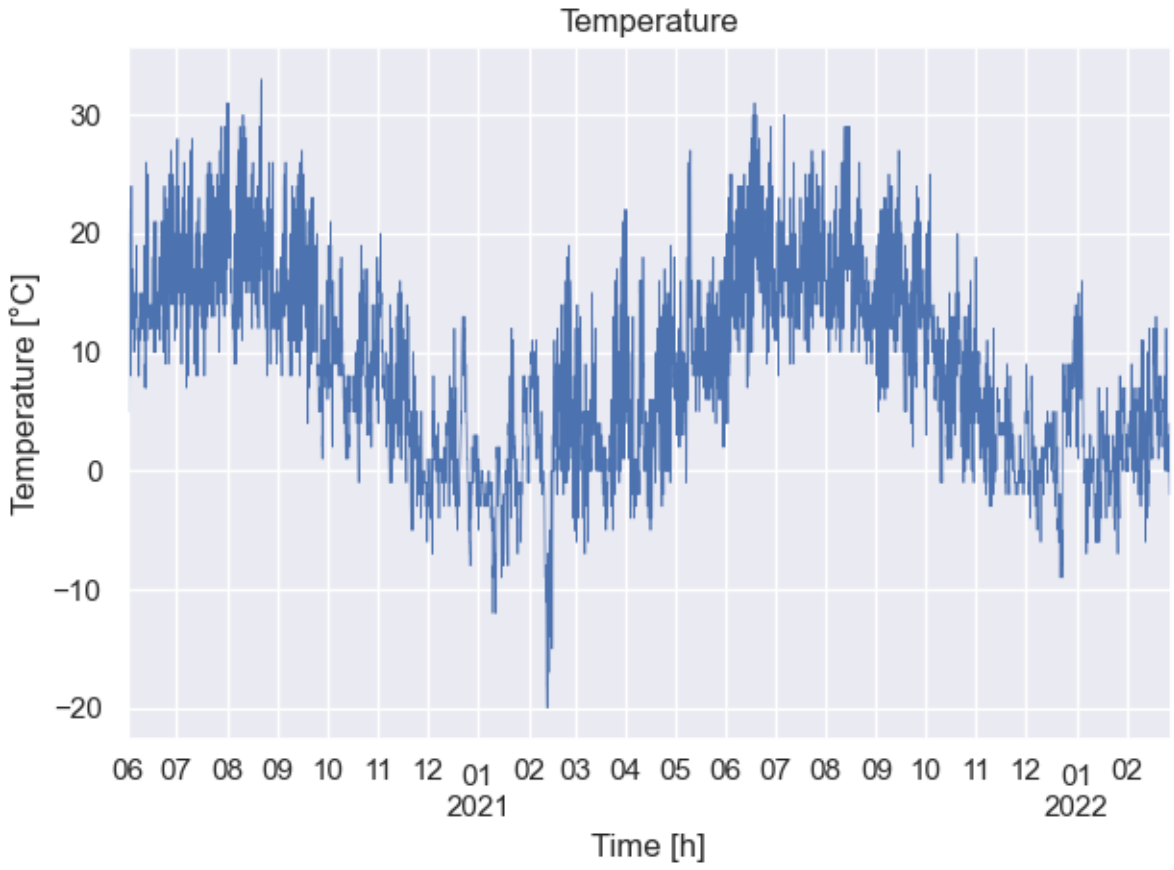


Figure 2.4: The hourly temperature measured in degrees Celsius [°C] from 1. June 2020, to 28. February 2022.

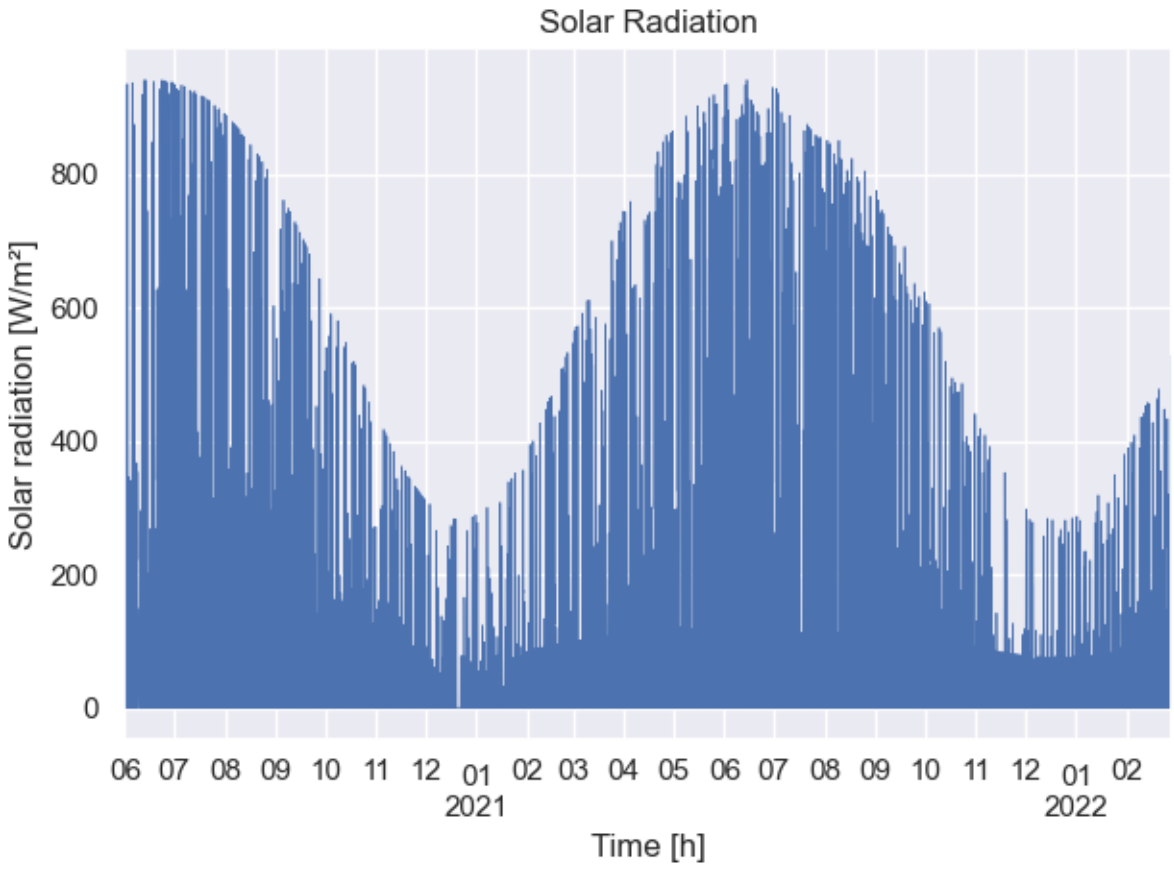


Figure 2.5: The hourly solar radiation measured in watt per square meter [W/m²] from 1. June 2020, to 28. February 2022.

Section 3: Main software libraries.

Software librarie	Version	Notes
Python[3]	3.8	The implementation of the code was done with Python version 3.8.
statsmodels[4]	0.13.2	The Python modue statsmodels is used for the analysis of the time series models.
Scikit-learn[5]	1.0.2	The implementation of the regression models was done with the software library Scikit-learn.
Keras-TensorFlow [6,7]	2.8.0	The ANNs were created using the Keras-TensorFlow interface.
padasip[8]	1.2.1	The implementation of the RLS is adapted and modified by the module padasip.
Git repository LIBOL/ ODL[9]	SHA: 5b981b2a08f359d7e 20444d4a31c43832ce58aa0	The implementation for the HBP was taken from the Git repository LIBOL/ ODL and adapted for the regression problem.
matplotlib[10] seaborn[11]	3.5.1 0.11.2	The figures were plotted with matplotlib and seaborn.

[3] Python Software Foundation. Python Language Reference, version 3.8. Available at http://www.python.org

[4] Seabold, Skipper, and Josef Perktold. "statsmodels: Econometric and statistical modeling with python." *Proceedings of the 9th Python in Science Conference.* 2010.

[5] Scikit-learn: Machine Learning in Python, Pedregosa *et al.*, JMLR 12, pp. 2825-2830, 2011.

[6] Chollet, F. (2015) keras, GitHub. https://github.com/fchollet/keras

[7] Martín Abadi et al., TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from https://www.tensorflow.org/ .

[8] https://pypi.org/project/padasip/

[9] https://github.com/LIBOL/ODL

[10] J. D. Hunter, "Matplotlib: A 2D Graphics Environment", Computing in Science & Engineering, vol. 9, no. 3, pp. 90-95, 2007.

[11] Waskom, M. et al., 2017. *mwaskom/seaborn: v0.8.1 (September 2017)*, Zenodo. Available at: https://doi.org/10.5281/zenodo.883859.

Section 4: Parameterization of the models.

This section lists the examined hyperparameter spaces for all parameterized models, and the optimal hyperparameter configuration found. The parameters of a model are optimized for the different seasons. for the summer months, for the winter months, and for the entire year.

4.1 Holt-Winter Smoothing (HW)

Hyperparameters	Search Space
trend	additive, multiplicative, None
seasonal	additive, multiplicative, None

Table 4.1.1: The examined hyperparameter space of the HW models. The optimization is performed with a grid search.

HW Model	Hyperparameters
summer	trend=None, seasonal=additive, seasonal_periods=24
winter	trend=multiplicative, seasonal=additive, seasonal_periods=24
all (entire year)	trend=None, seasonal=additive, seasonal_periods=24

Table 4.1.2: The optimal hyperparameters found for the HW models.

4.2 Seasonal Autoregressive Integrated Moving Average (SARIMA)

Hyperparameters	Search Space
p	0, 1, 2
d	0, 1
q	0, 1, 2
trend	n, c, t, ct
P	0, 1, 2
D	0, 1
Q	0, 1, 2

Table 4.2.1: The examined hyperparameter space of the SARIMA models. The optimization is performed with a grid search.

SARIMA Model	Hyperparameters
summer	order: p=0, d=0, q=0; trend=c; seasonal_order: P=2, D=0, Q=0, m=24
winter	order: p=0, d=0, q=0; trend=c; seasonal_order: P=2, D=0, Q=0, m=24
all	order: p=0, d=1, q=2; trend=c; seasonal_order: P=2, D=0, Q=0, m=24

Table 4.2.2: The optimal hyperparameters found for the SARIMA models.

4.3 Kernel Ridge Regression (KRR)

Hyperparameters	Search Space
kernel	rbf, linear
alpha	0.1, 0.2, 0.3, …, 9.8, 9.9, 10.0
gamma	0.1, 0.2, 0.3, …, 9.8, 9.9, 10.0

Table 4.3.1: The examined hyperparameter space of the KRR models. The optimization is performed with Bayesian optimization.

KRR Model	Hyperparameters
summer	kernel=rbf, alpha=0.1, gamma=3.2
winter	kernel=rbf, alpha=0.1, gamma=10.0
all	kernel=rbf, alpha=0.1, gamma=10.0

Table 4.3.2: The optimal hyperparameters found for the KRR models.

4.4 Support Vector Regression (SVR)

Hyperparameters	Search Space
kernel	rbf, linear
C	summer: 100, 110, 120, …, 46480, 46490, 46500 winter: 200, 300, 400, …, 104800, 104900, 105000 all: 100, 200, 300, …, 104800, 104900, 105000
gamma	auto, scale
epsilon	0.1, 0.2, 0.3, …, 99.8, 99.9, 100.0

Table 4.4.1: The examined hyperparameter space of the SVR models. The optimization is performed with Bayesian optimization.

SVR Model	Hyperparameters
summer	kernel=rbf, C=46500, gamma=scale, epsilon=100.0
winter	kernel=rbf, C=105000, gamma=scale, epsilon=52.2
all	kernel=rbf, C=105000, gamma=scale, epsilon=100.0

Table 4.4.2: The optimal hyperparameters found for the SVR models.

4.5 Random Forest Regression (RFR)

Hyperparameters	Search Space
n_estimators	100, 110, 120, …, 4980, 4990, 5000
max_features	auto, sqrt, log2
min_samples_split	2, 3, 4, …, 8, 9, 10
min_samples_leaf	2, 3, 4, …, 8, 9, 10
min_weight_fraction_leaf	0.0, 0.1, 0.2, 0.3, 0.4, 0.5
bootstrap	false (0), true (1)

Table 4.5.1: The examined hyperparameter space of the RFR models. The optimization is performed with Bayesian optimization.

RFR Model	Hyperparameters
summer	n_estimators=5000, max_features=log2, min_samples_split=2, min_samples_leaf=2, min_weight_fraction_leaf=0.0, bootstrap=1, max_depth=None
winter	n_estimators=5000, max_features=log2, min_samples_split=2, min_samples_leaf=2, min_weight_fraction_leaf=0.0, bootstrap=0, max_depth=None
all	n_estimators=5000, max_features=log2, min_samples_split=6, min_samples_leaf=2, min_weight_fraction_leaf=0.0, bootstrap=0, max_depth=None

Table 4.5.2: The optimal hyperparameters found for the RFR models.

4.6 Fully Connected Neural Network (DNN)

Hyperparameters	Search Space
batch_size	16, 32, 64, 128, 256
units	8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096
optimizer	sgd, rmsprop, adam
activation	linear, relu
dropout_rate	0.0, 0.1, 0.2, 0.3, 0.4, 0.5
kernel_initializer	lecun_uniform, he_normal, he_uniform, glorot_normal, glorot_uniform, normal, zero, ones, uniform, random_normal, random_uniform
regularizer	l1, l2, l1_l2
constraint	unit_norm, non_neg, max_norm, min_max_norm

Table 4.6.1: The examined hyperparameter space of the DNN models. The optimization is performed with Bayesian optimization.

DNN Model	Hyperparameters
summer	batch_size=64, optimizer=adam, epoch=1000, units_1=4096, activation_1=relu, kernel_regularizer_1=l1_l2, activity_regularizer_1=l1_l2, bias_regularizer_1=l1, kernel_initializer_1=random_uniform, kernel_constraint_1=unit_norm, bias_constraint_1=non_neg, dropout_1=0.4, units_2=4096, activation_2=linear, kernel_regularizer_2=l1, activity_regularizer_2=l1_l2, bias_regularizer_2=l1, kernel_initializer_2=random_uniform, kernel_constraint_2=unit_norm, bias_constraint_2=min_max_norm, dropout_2=0.4, units_3=1, activation_3=relu, kernel_regularizer_3=l1_l2, activity_regularizer_3=l1_l2, bias_regularizer_3=l1_l2, kernel_initializer_3=he_uniform, kernel_constraint_3=non_neg, bias_constraint_3=min_max_norm
winter	batch_size=128, optimizer=adam, epoch=1000, units_1=2048, activation_1=relu, kernel_regularizer_1=l1_l2, activity_regularizer_1=l2, bias_regularizer_1=l1, kernel_initializer_1=lecun_uniform, kernel_constraint_1=min_max_norm, bias_constraint_1=min_max_norm, dropout_1=0.0, units_2=4096, activation_2=linear, kernel_regularizer_2=l1_l2, activity_regularizer_2=l1, bias_regularizer_2=l1, kernel_initializer_2=random_uniform, kernel_constraint_2=non_neg, bias_constraint_2=min_max_norm, dropout_2=0.1, units_3=1, activation_3=relu, kernel_regularizer_3=l1_l2, activity_regularizer_3=l1_l2, bias_regularizer_3=l1, kernel_initializer_3=lecun_uniform, kernel_constraint_3=non_neg, bias_constraint_3=unit_norm
all	batch_size=16, optimizer=adam, epochs=1000, units_1=4096, activation_1=linear, kernel_regularizer_1=l1, activity_regularizer_1=l1, bias_regularizer_1=l1_l2, kernel_initializer_1=random_uniform, kernel_constraint_1=min_max_norm, bias_constraint_1=unit_norm, dropout_1=0.5, units_2=4096, activation_2=relu, kernel_regularizer_2=l1_l2, activity_regularizer_2=l1, bias_regularizer_2=l1_l2, kernel_initializer_2=lecun_uniform, kernel_constraint_2=non_neg, bias_constraint_2=min_max_norm, dropout_2=0.5, units_3=1, activation_3=relu, kernel_regularizer_3=l1_l2, activity_regularizer_3=l1, bias_regularizer_3=l1_l2, kernel_initializer_3=glorot_uniform, kernel_constraint_3=min_max_norm, bias_constraint_3=unit_norm

Table 4.6.2: The optimal hyperparameters found for the DNN models.

4.7 Convolutional Neural Networks (CNN)

Hyperparameters	Search Space
batch_size	16, 32, 64, 128, 256
units	8, 16, 32, 64, 128, 256, 512, 1024, 2048
optimizer	sgd, rmsprop, adam
activation	linear, relu
filters	4, 8, 16, 32, 64, 128
kernel_size	3, 5, 7, 9
dropout_rate	0.0, 0.1, 0.2, 0.3, 0.4, 0.5
kernel_initializer	lecun_uniform, he_normal, he_uniform, glorot_normal, glorot_uniform, normal, zero, ones, uniform
recurrent_initializer	orthogonal, lecun_uniform, he_normal, he_uniform, glorot_normal, glorot_uniform, normal
regularizer	l1, l2, l1_l2
constraint	unit_norm, non_neg, max_norm, min_max_norm

Table 4.7.1: The examined hyperparameter space of the CNN models. The optimization is performed with Bayesian optimization.

CNN Model	Hyperparameters
summer	batch_size=16, optimizer=adam, epochs=1000, filter_cnn_1=128, kernel_size_cnn_1=9, activation_cnn_1=relu, kernel_regularizer_cnn_1=l2, activity_regularizer_cnn_1=l1_l2, bias_regularizer_cnn_1=l1_l2, kernel_initializer_cnn_1=uniform, kernel_constraint_cnn_1=max_norm, bias_constraint_cnn_1=min_max_norm, dropout_1=0.1, filter_cnn_2=64, kernel_size_cnn_2=3, activation_cnn_2=relu, kernel_regularizer_cnn_2=l2, activity_regularizer_cnn_2=l1, bias_regularizer_cnn_2=l1_l2, kernel_initializer_cnn_2=glorot_normal, kernel_constraint_cnn_2=unit_norm, bias_constraint_cnn_2=max_norm, dropout_2=0.1, filter_cnn_3=128, kernel_size_cnn_3=3,vactivation_cnn_3=relu, kernel_regularizer_cnn_3=l1, activity_regularizer_cnn_3=l1, bias_regularizer_cnn_3=l2, kernel_initializer_cnn_3=uniform, kernel_constraint_cnn_3=unit_norm, bias_constraint_cnn_3=max_norm, dropout_3=0.0, units_dense_1=16, activation_dense_1=relu, kernel_regularizer_dense_1=l1, activity_regularizer_dense_1=l1_l2, bias_regularizer_dense_1=l1, kernel_initializer_dense_1=normal, kernel_constraint_dense_1=unit_norm, bias_constraint_dense_1=min_max_norm, units_dense_2=1, activation_dense_2=relu, kernel_regularizer_dense_2=l1, activity_regularizer_dense_2=l1_l2, bias_regularizer_dense_2=l1_l2, kernel_initializer_dense_2=uniform, kernel_constraint_dense_2=max_norm, bias_constraint_dense_2=non_neg
winter	batch_size=16, optimizer=nadam, filter_cnn_1=64, kernel_size_cnn_1=9, activation_cnn_1=relu, kernel_regularizer_cnn_1=l1, activity_regularizer_cnn_1=l1_l2, bias_regularizer_cnn_1=l1_l2, kernel_initializer_cnn_1=lecun_uniform, kernel_constraint_cnn_1=min_max_norm, bias_constraint_cnn_1=non_neg, dropout_1=0.0, filter_cnn_2=128, kernel_size_cnn_2=5, activation_cnn_2=linear, kernel_regularizer_cnn_2=l1_l2, activity_regularizer_cnn_2=l1, bias_regularizer_cnn_2=l1_l2, kernel_initializer_cnn_2=he_uniform, kernel_constraint_cnn_2=max_norm, bias_constraint_cnn_2=min_max_norm, dropout_2=0.0, filter_cnn_3=128, kernel_size_cnn_3=9, activation_cnn_3=relu, kernel_regularizer_cnn_3=l1, activity_regularizer_cnn_3=l1, bias_regularizer_cnn_3=l1_l2, kernel_initializer_cnn_3=lecun_uniform, kernel_constraint_cnn_3=unit_norm, bias_constraint_cnn_3=min_max_norm, dropout_3=0.0, units_dense_1=128, activation_dense_1=relu, kernel_regularizer_dense_1=l2, activity_regularizer_dense_1=l2, bias_regularizer_dense_1=l1, kernel_initializer_dense_1=lecun_uniform, kernel_constraint_dense_1=non_neg, bias_constraint_dense_1=max_norm, units_dense_2=1, activation_dense_2=relu, kernel_regularizer_dense_2=l1, activity_regularizer_dense_2=l1_l2, bias_regularizer_dense_2=l1_l2, kernel_initializer_dense_2=he_uniform, kernel_constraint_dense_2=unit_norm, bias_constraint_dense_2=non_neg
all	batch_size=16, optimizer=adam, epochs=1000, filter_cnn_1=16, kernel_size_cnn_1=9, activation_cnn_1=relu, kernel_regularizer_cnn_1=l1, activity_regularizer_cnn_1=l2, bias_regularizer_cnn_1=l1_l2, kernel_initializer_cnn_1=lecun_uniform, kernel_constraint_cnn_1=min_max_norm, bias_constraint_cnn_1=non_neg, dropout_1=0.0, filter_cnn_2=64, kernel_size_cnn_2=5,activation_cnn_2=linear, kernel_regularizer_cnn_2=l1, activity_regularizer_cnn_2=l1, bias_regularizer_cnn_2=l2, kernel_initializer_cnn_2=lecun_uniform, kernel_constraint_cnn_2=non_neg, bias_constraint_cnn_2=max_norm, dropout_2=0.2, filter_cnn_3=128, kernel_size_cnn_3=9, activation_cnn_3=relu, kernel_regularizer_cnn_3=l1, activity_regularizer_cnn_3=l1, bias_regularizer_cnn_3=l2, kernel_initializer_cnn_3=lecun_uniform, kernel_constraint_cnn_3=unit_norm, bias_constraint_cnn_3=min_max_norm, dropout_3=0.0, units_dense_1=2048, activation_dense_1=linear, kernel_regularizer_dense_1=l1_l2, activity_regularizer_dense_1=l1, bias_regularizer_dense_1=l1_l2, kernel_initializer_dense_1=lecun_uniform, kernel_constraint_dense_1=unit_norm, bias_constraint_dense_1=max_norm, units_dense_2=1, activation_dense_2=relu, kernel_regularizer_dense_2=l1, activity_regularizer_dense_2=l2, bias_regularizer_dense_2=l1_l2, kernel_initializer_dense_2=lecun_uniform, kernel_constraint_dense_2=unit_norm, bias_constraint_dense_2=non_neg

Table 4.7.2: The optimal hyperparameters found for the CNN models.

4.8 Long Short-Term Memory (LSTM)

Hyperparameters	Search Space
batch_size	16, 32, 64, 128, 256
units	8, 16, 32, 64, 128, 256, 512, 1024, 2048
optimizer	sgd, rmsprop, adam
activation	linear, relu
lstm_activation	linear, relu, tanh, sigmoid
dropout_rate	0.0, 0.1, 0.2, 0.3, 0.4, 0.5
kernel_initializer	lecun_uniform, he_normal, he_uniform, glorot_normal, glorot_uniform, normal, zero, ones, uniform
recurrent_initializer	orthogonal, lecun_uniform, he_normal, he_uniform, glorot_normal, glorot_uniform, normal
regularizer	l1, l2, l1_l2
Constraint	unit_norm, non_neg, max_norm, min_max_norm
use_bias	false (0), true (1)
unit_forget_bias	false (0), true (1)
go_backwards	false (0), true (1)
time_major	false (0), true (1)

Table 4.8.1: The examined hyperparameter space of the LSTM models. The optimization is performed with Bayesian optimization.

LSTM Model	Hyperparameters
summer	batch_size=16, optimizer=adam, epochs=1000, units_lstm=512, activation_lstm=sigmoid, recurrent_activation_lstm=linear, use_bias_lstm=0, dropout_lstm=0.1, recurrent_dropout_lstm=0.3, kernel_initializer_lstm=uniform, recurrent_initializer_lstm=lecun_uniform, unit_forget_bias_lstm=0, recurrent_regularizer_lstm=l1_l2, kernel_regularizer_lstm=l2, bias_regularizer_lstm=l1_l2, activity_regularizer_lstm=l1_l2, kernel_constraint_lstm=non_neg, bias_constraint_lstm=min_max_norm, recurrent_constraint_lstm=unit_norm, go_backwards_lstm=1, time_major_lstm=0, units_dense_1=32, activation_dense_1=relu, kernel_regularizer_dense_1=l2, activity_regularizer_dense_1=l1_l2, bias_regularizer_dense_1=l1_l2, kernel_initializer_dense_1=he_uniform, kernel_constraint_dense_1=non_neg, bias_constraint_dense_1=unit_norm, dropout_1=0.0, units_dense_2=1, activation_dense_2=relu, kernel_regularizer_dense_2=l2, activity_regularizer_dense_2=l1_l2, bias_regularizer_dense_2=l1_l2, kernel_initializer_dense_2=glorot_normal, kernel_constraint_dense_2=max_norm, bias_constraint_dense_2=non_neg
winter	batch_size=16, optimizer=adam, epochs=1000, units_lstm=64, activation_lstm=linear, recurrent_activation_lstm=relu, use_bias_lstm=1, dropout_lstm=0.2, recurrent_dropout_lstm=0.1, kernel_initializer_lstm=glorot_uniform, recurrent_initializer_lstm=glorot_uniform, unit_forget_bias_lstm=1, recurrent_regularizer_lstm=l1, kernel_regularizer_lstm=l1_l2, bias_regularizer_lstm=l1, activity_regularizer_lstm=l1, kernel_constraint_lstm=min_max_norm, bias_constraint_lstm=non_neg, recurrent_constraint_lstm=max_norm, go_backwards_lstm=0, time_major_lstm=0, units_dense_1=32, activation_dense_1=relu, kernel_regularizer_dense_1=l1, activity_regularizer_dense_1=l2, bias_regularizer_dense_1=l2, kernel_initializer_dense_1=he_normal, kernel_constraint_dense_1=non_neg, bias_constraint_dense_1=min_max_norm, dropout_1=0.4, units_dense_2=1, activation_dense_2=relu, kernel_regularizer_dense_2=l1_l2, activity_regularizer_dense_2=l2, bias_regularizer_dense_2=l1, kernel_initializer_dense_2=glorot_normal, kernel_constraint_dense_2=max_norm, bias_constraint_dense_2=non_neg

all	batch_size=16, optimizer=adam, epochs=1000, units_lstm=512, activation_lstm=sigmoid, recurrent_activation_lstm=linear, use_bias_lstm=0, dropout_lstm=0.1, recurrent_dropout_lstm=0.3, kernel_initializer_lstm=uniform, recurrent_initializer_lstm=lecun_uniform, unit_forget_bias_lstm=0, recurrent_regularizer_lstm=l1_l2, kernel_regularizer_lstm=l2, bias_regularizer_lstm=l1_l2, activity_regularizer_lstm=l1_l2, kernel_constraint_lstm=non_neg, bias_constraint_lstm=min_max_norm, recurrent_constraint_lstm=unit_norm, go_backwards_lstm=1, time_major_lstm=0, units_dense_1=32, activation_dense_1=relu, kernel_regularizer_dense_1=l2, activity_regularizer_dense_1=l1_l2, bias_regularizer_dense_1=l1_l2, kernel_initializer_dense_1=he_uniform, kernel_constraint_dense_1=non_neg, bias_constraint_dense_1=unit_norm, dropout_1=0.0, units_dense_2=1, activation_dense_2=relu, kernel_regularizer_dense_2=l2, activity_regularizer_dense_2=l1_l2, bias_regularizer_dense_2=l1_l2, kernel_initializer_dense_2=glorot_normal, kernel_constraint_dense_2=max_norm, bias_constraint_dense_2=non_neg

Table 4.8.2: The optimal hyperparameters found for the LSTM models.

4.9 Stochastic Gradient Descent (SGD)

Hyperparameters	**Search Space**
loss	huber, squared_loss, epsilon_insensitive, squared_epsilon_insensitive
penalty	l2, l1, elasticnet
learning_rate	invscaling, optimal, constant, adaptive

Table 4.9.1: The examined hyperparameter space of the SGD models. The optimization is performed with a grid search.

SGD Model	**Hyperparameters**
summer	loss=huber, penalty=l1, learning_rate=invscaling
winter	loss=huber, penalty=l1, learning_rate=invscaling
all	loss=huber, penalty=l2, learning_rate=invscaling

Table 4.9.2: The optimal hyperparameters found for the SGD models.

4.10 Recursive Least Squares (RLS)

Hyperparameters	**Search Space**
mu	0.01, 0.02, 0.03, …, 0.98, 0.99, 1.0
epsilon	0.01, 0.02, 0.03, …, 0.98, 0.99, 1.0

Table 4.10.1: The examined hyperparameter space of the RLS models. The optimization is performed with a grid search.

RLS Model	**Hyperparameters**
summer	mu=0.99, epsilon=0.01
winter	mu=0.99, epsilon=0.01
all	mu=0.99, epsilon=0.01

Table 4.10.2: The optimal hyperparameters found for the RLS models.

4.11 Online Deep Learning (ODL)

Hyperparameters	Search Space
units	8, 16, 32, 64, 128, 256, 512, 1024, 2048
optimizer	sgd, rmsprop, adam
activation	linear, relu
learning_rate	1e-3, 1e-4, 1e-5, 1e-6
kernel_initializer	lecun_uniform, he_normal, he_uniform, glorot_normal, glorot_uniform, normal, zero, ones, uniform, random_normal, random_uniform
regularizer	l1, l2, l1_l2
constraint	unit_norm, non_neg, max_norm, min_max_norm

Table 4.11.1: The examined hyperparameter space of the ODL models. The optimization is performed with Bayesian optimization

ODL Model	Hyperparameters
summer	optimizer=adam, learning_rate=0.001, epochs=50, units_1=512, activation_1=relu, kernel_regularizer_1=l2, activity_regularizer_1=l1, bias_regularizer_1=l1, kernel_initializer_1=random_uniform, kernel_constraint_1=non_neg, bias_constraint_1=min_max_norm, units_2=512, activation_2=linear, kernel_regularizer_2=l2, activity_regularizer_2=l2, bias_regularizer_2=l2, kernel_initializer_2=normal, kernel_constraint_2=min_max_norm, bias_constraint_2=unit_norm, units_3=1, activation_3=relu, kernel_regularizer_3=l2, activity_regularizer_3=l1_l2, bias_regularizer_3=l1_l2, kernel_initializer_3=glorot_uniform, kernel_constraint_3=unit_norm, bias_constraint_3=max_norm
winter	optimizer=adam, learning_rate=0.001, epochs=50, units_1=512, activation_1=relu, kernel_regularizer_1=l2, activity_regularizer_1=l1, bias_regularizer_1=l1, kernel_initializer_1=random_uniform, kernel_constraint_1=non_neg, bias_constraint_1=min_max_norm, units_2=512, activation_2=linear, kernel_regularizer_2=l2, activity_regularizer_2=l2, bias_regularizer_2=l2, kernel_initializer_2=normal, kernel_constraint_2=min_max_norm, bias_constraint_2=unit_norm, units_3=1, activation_3=relu, kernel_regularizer_3=l2, activity_regularizer_3=l1_l2, bias_regularizer_3=l1_l2, kernel_initializer_3=glorot_uniform, kernel_constraint_3=unit_norm, bias_constraint_3=max_norm
all	optimizer=adam, learning_rate=0.001, epochs=50, units_1=512, activation_1=relu, kernel_regularizer_1=l2, activity_regularizer_1=l1, bias_regularizer_1=l1, kernel_initializer_1=random_uniform, kernel_constraint_1=non_neg, bias_constraint_1=min_max_norm, units_2=512, activation_2=linear, kernel_regularizer_2=l2, activity_regularizer_2=l2, bias_regularizer_2=l2, kernel_initializer_2=normal, kernel_constraint_2=min_max_norm, bias_constraint_2=unit_norm, units_3=1, activation_3=relu, kernel_regularizer_3=l2, activity_regularizer_3=l1_l2, bias_regularizer_3=l1_l2, kernel_initializer_3=glorot_uniform, kernel_constraint_3=unit_norm, bias_constraint_3=max_norm

Table 4.11.2: The optimal hyperparameters found for the ODL models.

4.12 ODL with an Experience Replay (ODL-ER)

For the neural network, the optimal found parameters of the ODL are used. The parameters for the Experience Replay (ER) are optimized.

Hyperparameters	Search Space
batch_size	4, 8, 16
capacity	500, 1000, 5000, 10000, 20000
factor	0.1, 0.2, 0.3, 0.4, 0.5
min_recollection	0, 24, 48, 168, 336, 720

Table 4.12.1: The examined hyperparameter space of the ER models. The optimization is performed with a grid search.

ER Model	Hyperparameters
summer	batch_size=4, capacity=20000, factor=0.5, min_recollection=48, epochs=1000
winter	batch_size=4, capacity=10000, factor=0.5, min_recollection=48, epochs=1000
all	batch_size=16, capacity=20000, factor=0.5, min_recollection=0, epochs=1000

Table 4.12.2: The optimal hyperparameters found for the ER models.

4.13 Hedge Backpropagation (HBP)

Hyperparameters	Search Space
n_layers	2, 3, 4, …, 8, 9, 10
units	8, 16, 32, 64, 128, 256, 512, 1024, 2048
learning_rate	1e-3, 1e-4, 1e-5
activation	linear, relu
kernel_initializer	lecun_uniform, he_normal, he_uniform, glorot_normal, glorot_uniform, normal, zero, ones, uniform, random_normal, random_uniform
regularizer	l1, l2, l1_l2
constraint	unit_norm, non_neg, max_norm, min_max_norm

Table 4.13.1: The examined hyperparameter space of the HPB models. The optimization is performed with Bayesian optimization.

HBP Model	Hyperparameters
summer	n_layers=2, hidden_units=8, activation=linear, out_activation=relu, optimizer=adam, learning_rate=0.001, epochs=50, kernel_initializer=glorot_normal, kernel_regularizer=l1_l2, activity_regularizer=l1, bias_regularizer=l1_l2, kernel_constraint=non_neg, bias_constraint=min_max_norm
winter	n_layers=2, hidden_units=128, activation=relu, out_activation=relu, optimizer=adam, learning_rate=0.001, epochs=50, kernel_initializer=random_uniform, kernel_regularizer=l1_l2, activity_regularizer=l1, bias_regularizer=l1_l2, kernel_constraint=max_norm, bias_constraint=min_max_norm
all	n_layers=2, hidden_units=64, activation=relu, out_activation=relu, optimizer=adam, learning_rate=0.001, epochs=50, kernel_initializer=random_uniform, kernel_regularizer=l1_l2, activity_regularizer=l1, bias_regularizer=l1, kernel_constraint=min_max_norm, bias_constraint=min_max_norm

Table 4.13.2: The optimal hyperparameters found for the HBP models.

Section 5: Values of the prediction metrics of the online machine learning models.

The following tables show the values of the prediction metrics Root Mean Square Error (RMSE), Mean Absolute Scaled Error (MASE), Mean Absolute Deviation from the Median (MAD), and RMSE of the anomalies for the online machine learning methods. The online machine learning models are Recursive Least Squares (RLS), Stochastic Gradient Descent (SGD), Online Deep Learning (ODL), ODL with an Experience Replay (ODL-ER), and Hedge Backpropagation (HBP). The left column indicates the period from which the training and evaluation data are taken. Models trained and evaluated with the entire data set are referred to as "all". Models that have been trained and evaluated using data from the summer or winter months are referred to as "summer" or "winter". The upper row indicates the models investigated. The "24" refers to a forecast up to 24 hours ahead, while the "48" refers to a forecast from 25 to 48 hours. The values of the metrics are all measured in kilowatt-hours (kWh).

RMSE [kWh]	RLS-24	RLS-48	SGD-24	SGD-48	ODL-24	ODL-48	ODL-ER-24	ODL-ER-48	HBP-24	HBP-48
summer	4.16	4.15	5.13	5.17	4.42	4.33	3.83	3.84	4.29	4.18
winter	9.58	9.49	25.01	24.71	10.60	10.44	8.64	8.63	10.59	10.53
all	7.49	7.43	15.55	15.44	8.50	8.01	6.61	6.94	8.13	8.07
all-summer	4.15	4.13	5.07	5.07	4.37	4.28	4.15	4.15	4.29	4.19
all-winter	9.57	9.50	22.09	21.90	10.46	10.34	8.68	8.67	10.53	10.49

Table 5.1: Comparison of the RMSE values in kWh of the five online methods for the two forecast horizons 24 and 48 hours ahead and for the different period's summer, winter, and all year.

MAD [kWh]	RLS-24	RLS-48	SGD-24	SGD-48	ODL-24	ODL-48	ODL-ER-24	ODL-ER-48	HBP-24	HBP-48
summer	0.00	0.00	1.60	1.59	0.09	0.10	0.10	0.10	0.05	0.05
winter	0.00	0.00	18.59	18.47	0.19	0.19	0.31	0.31	0.20	0.20
all	0.00	0.00	10.06	9.96	0.15	0.15	0.20	0.20	0.12	0.12
all-summer	0.00	0.00	1.44	1.41	0.08	0.08	0.15	0.15	0.10	0.09
all-winter	0.00	0.00	15.44	15.29	0.19	0.19	0.23	0.23	0.13	0.13

Table 5.2: Comparison of the MAD values in kWh of the five online methods for the two forecast horizons 24 and 48 hours ahead and for the different period's summer, winter, and all year.

MASE [kWh]	RLS-24	RLS-48	SGD-24	SGD-48	ODL-24	ODL-48	ODL-ER-24	ODL-ER-48	HBP-24	HBP-48
summer	1.04	1.04	1.24	1.26	1.08	1.07	1.16	1.10	1.07	1.05
winter	1.44	1.42	4.15	4.13	1.58	1.56	1.24	1.03	1.60	1.60
all	1.29	1.28	2.70	2.70	1.37	1.36	1.23	1.07	1.39	1.38
all-summer	1.04	1.03	1.21	1.22	1.06	1.05	1.22	1.17	1.06	1.03
all-winter	1.44	1.43	3.63	3.61	1.56	1.55	1.23	1.01	1.59	1.59

Table 5.3: Comparison of the MASE values in kWh of the five online methods for the two forecast horizons 24 and 48 hours ahead and for the different period's summer, winter, and all year.

Anomaly (RMSE [kWh])	RLS-24	RLS-48	SGD-24	SGD-48	ODL-24	ODL-48	ODL-ER-24	ODL-ER-48	HBP-24	HBP-48
summer	6.27	6.06	9.62	8.41	8.00	6.25	5.25	5.23	7.21	5.71
winter	12.18	11.62	40.44	37.27	15.71	14.83	9.95	9.68	16.57	15.01
all	11.58	11.01	33.43	31.21	14.49	13.60	9.53	9.27	14.73	13.81
all-summer	5.88	5.83	9.71	8.52	7.58	6.16	5.34	5.35	6.46	5.77
all-winter	12.55	11.86	36.86	34.23	15.68	14.79	10.26	9.95	16.07	15.08

Table 5.4: Comparison of the RMSE values in kWh for the anomalies of the five online methods for the two forecast horizons 24 and 48 hours in advance and for the different period's summer, winter, and all year.

Section 6: Distribution of the RMSE values of the offline and online machine learning models[12]

The distribution of the RMSE values of the models trained and evaluated on the entire data are depicted. Count specifies the number of values that are in a bin. KDE refers to the kernel density estimation.

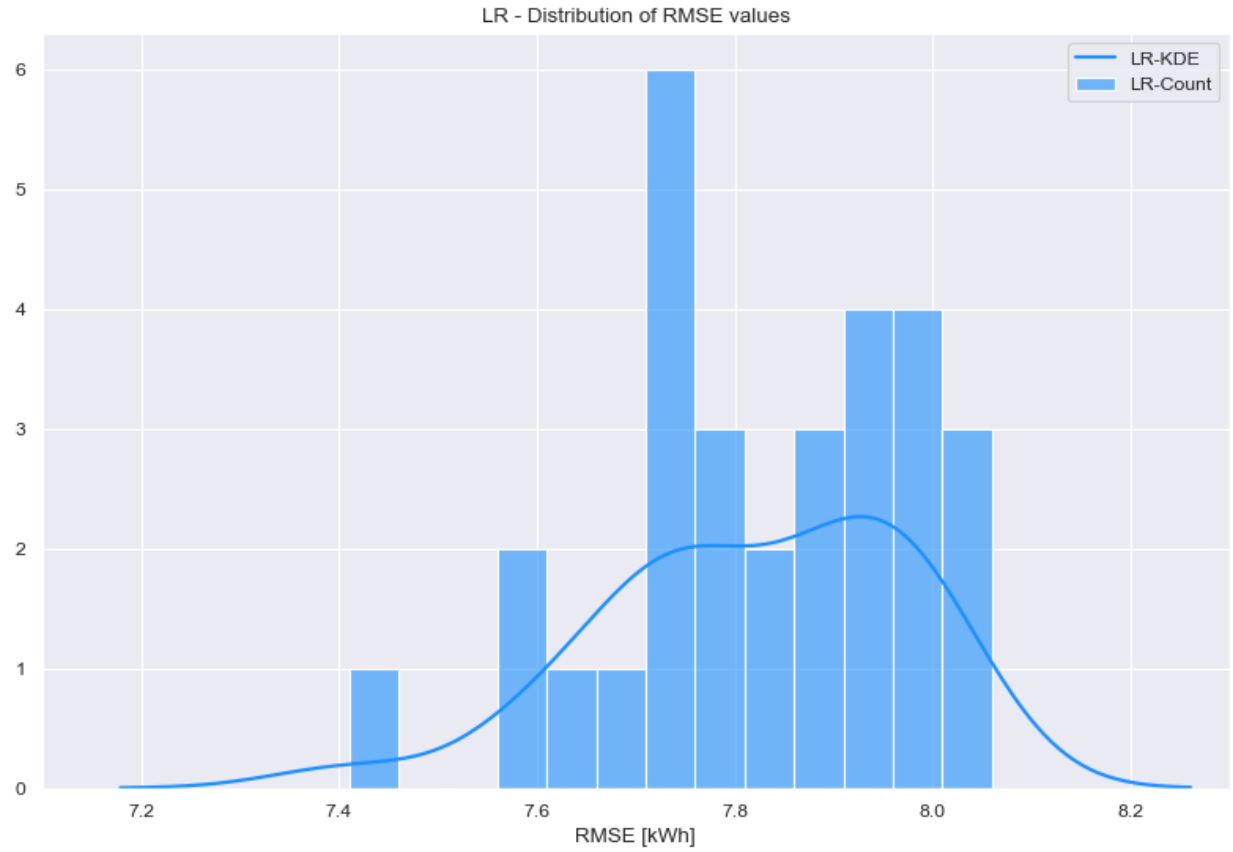


Figure 6.1: Distribution of RMSE values for LR.

[12] For the Recursive Least Squares (RLS), no distribution plot is calculated. The prediction or the errors of the RLS are calculated only once because it is a filter, and the forecast does not change even with repeated runs.

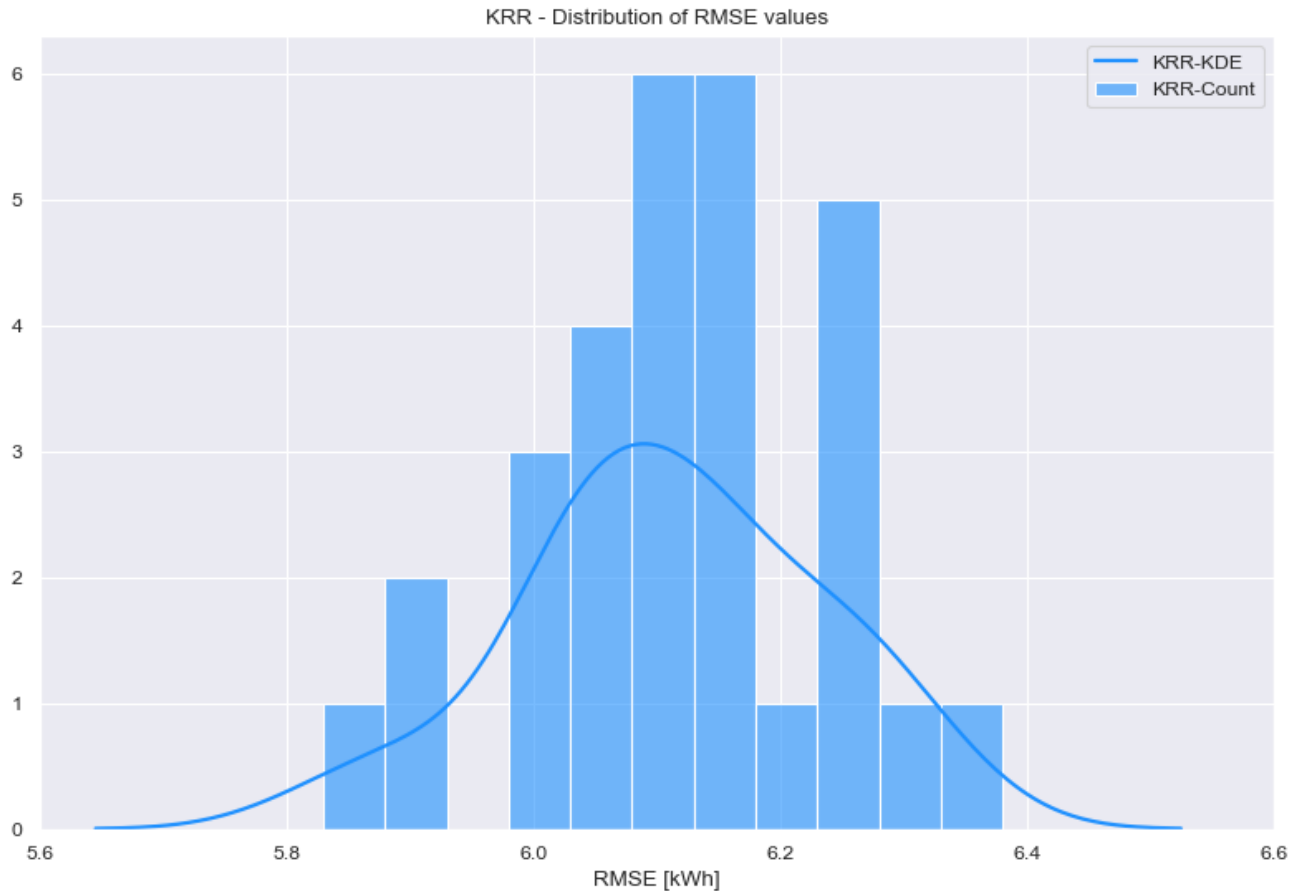


Figure 6.2: Distribution of RMSE values for KRR.

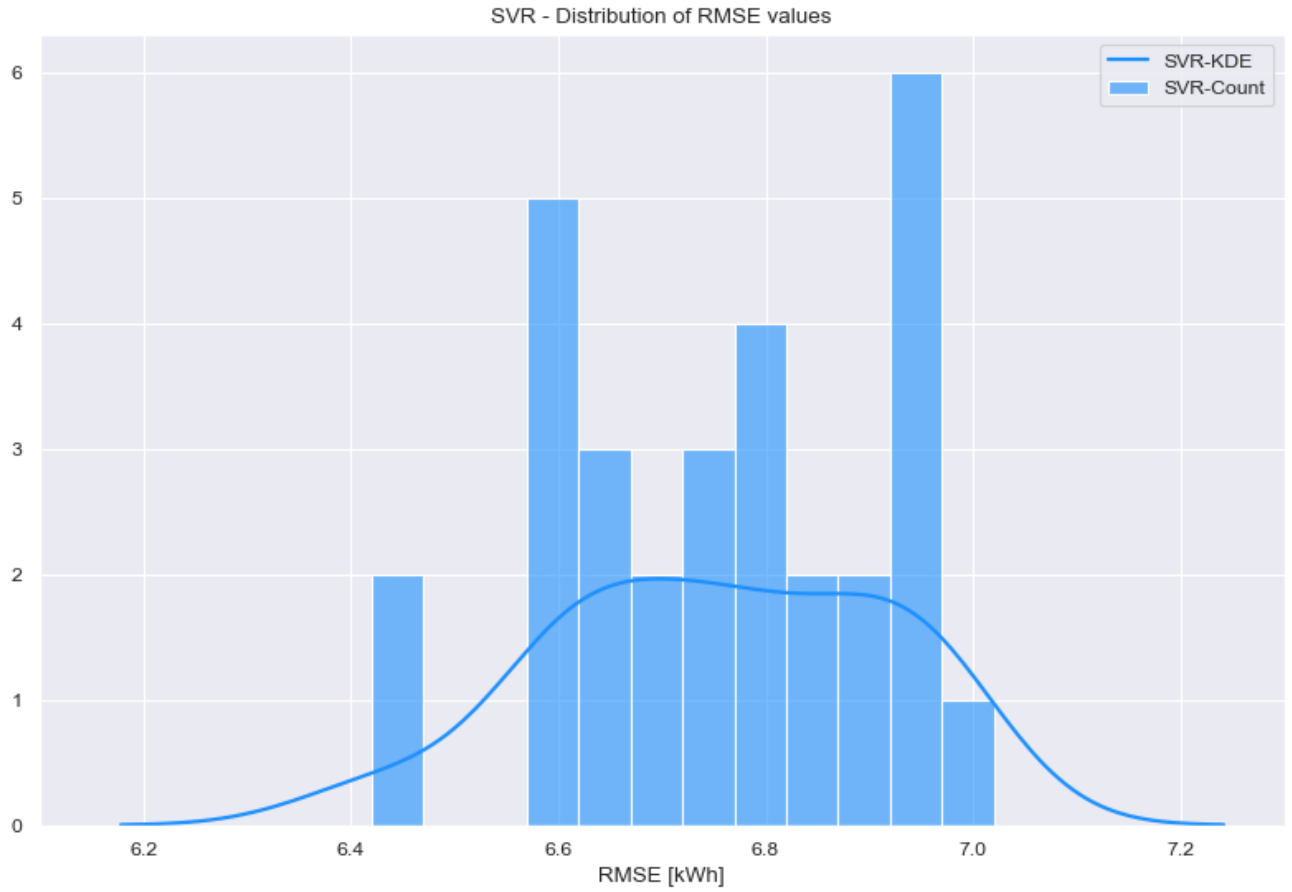


Figure 6.3: Distribution of RMSE values for SVR.

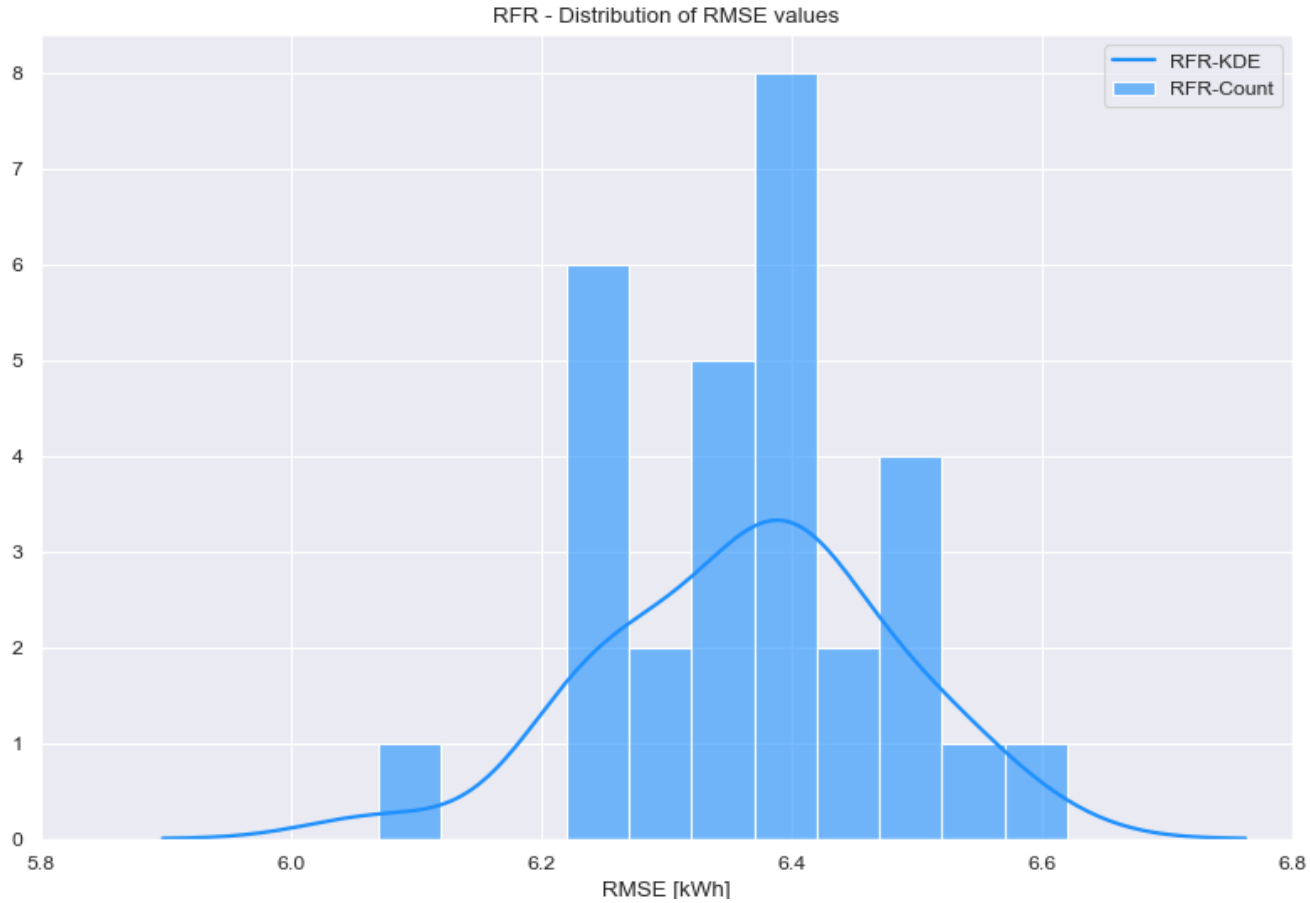


Figure 6.4: Distribution of RMSE values for RFR.

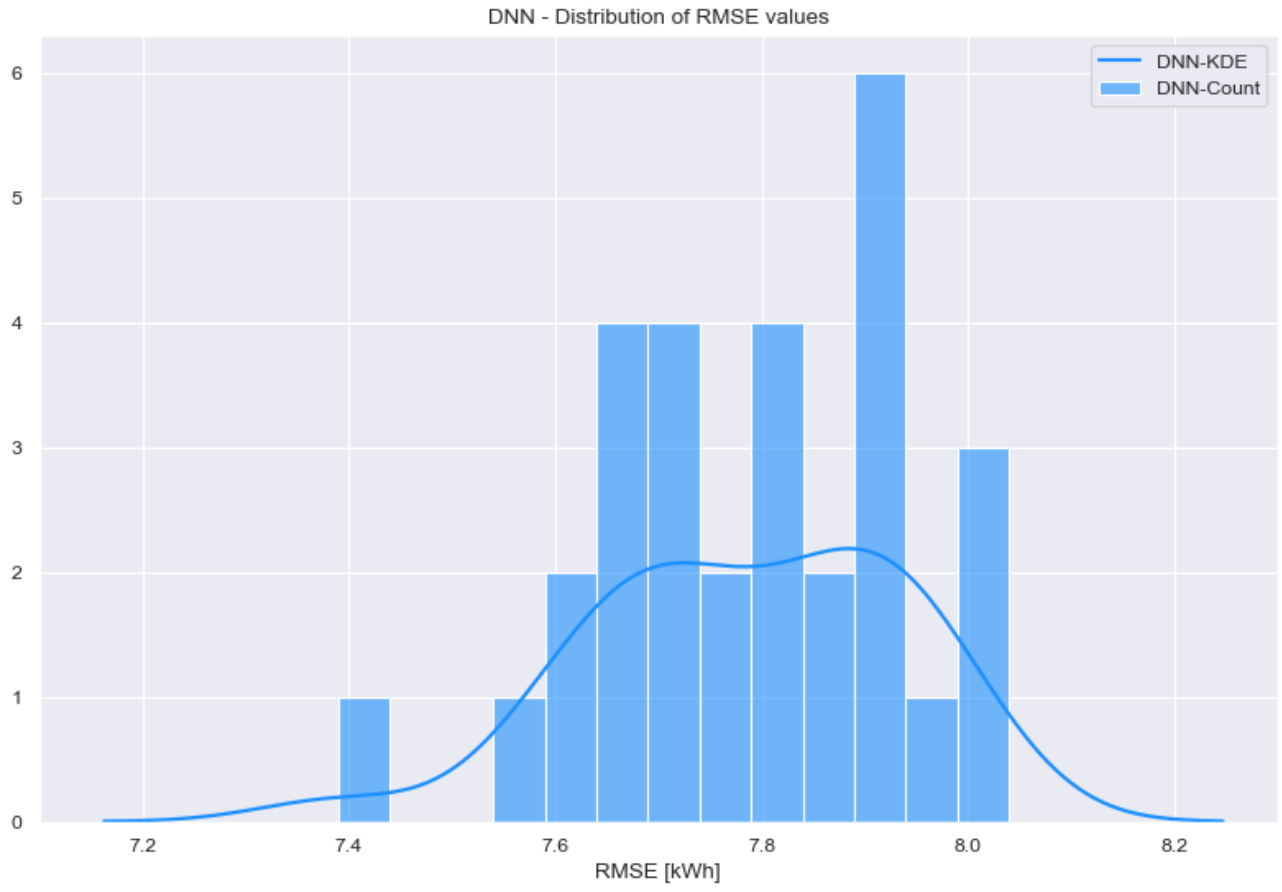


Figure 6.5: Distribution of RMSE values for DNN.

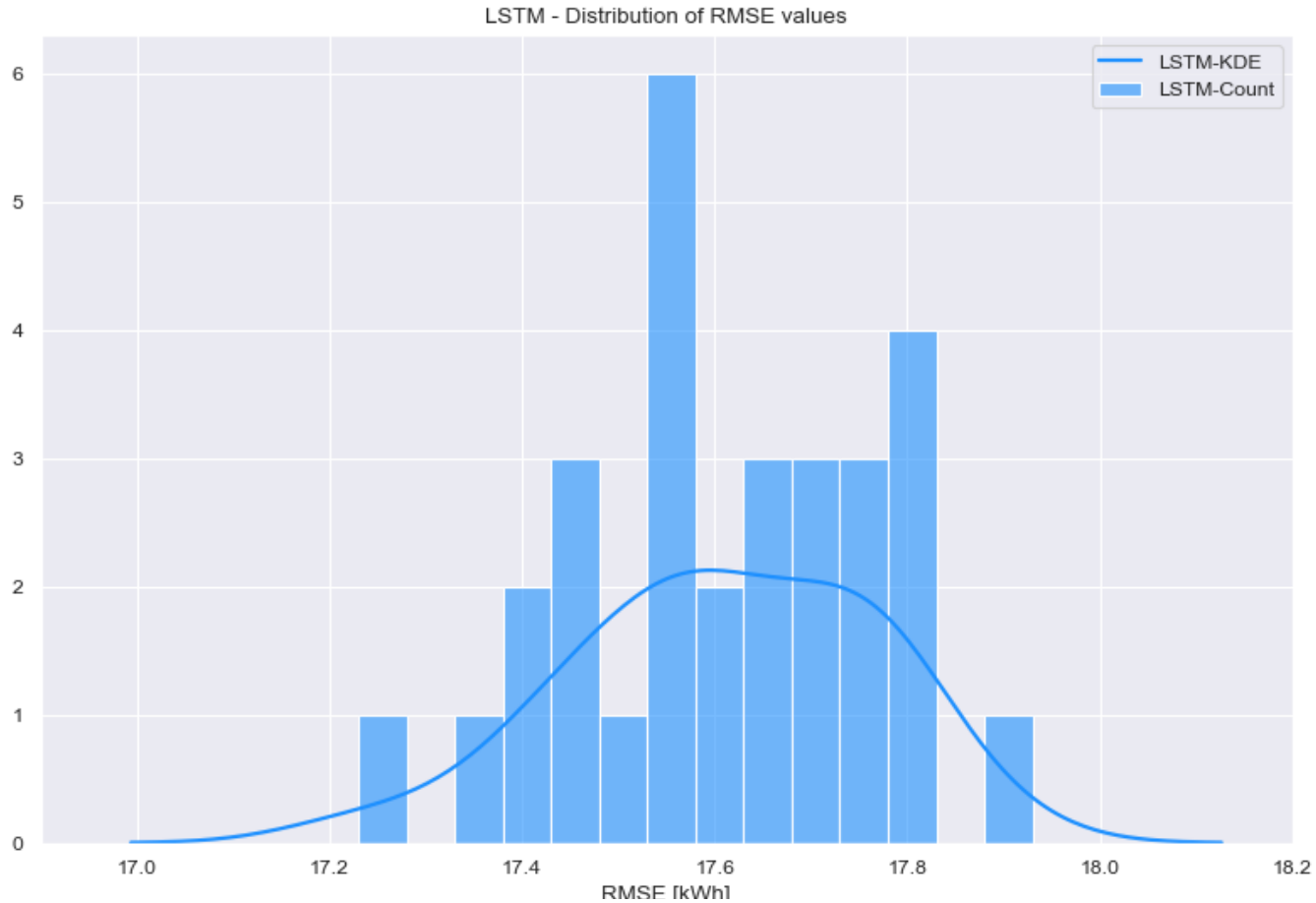


Figure 6.6: Distribution of RMSE values for LSTM.

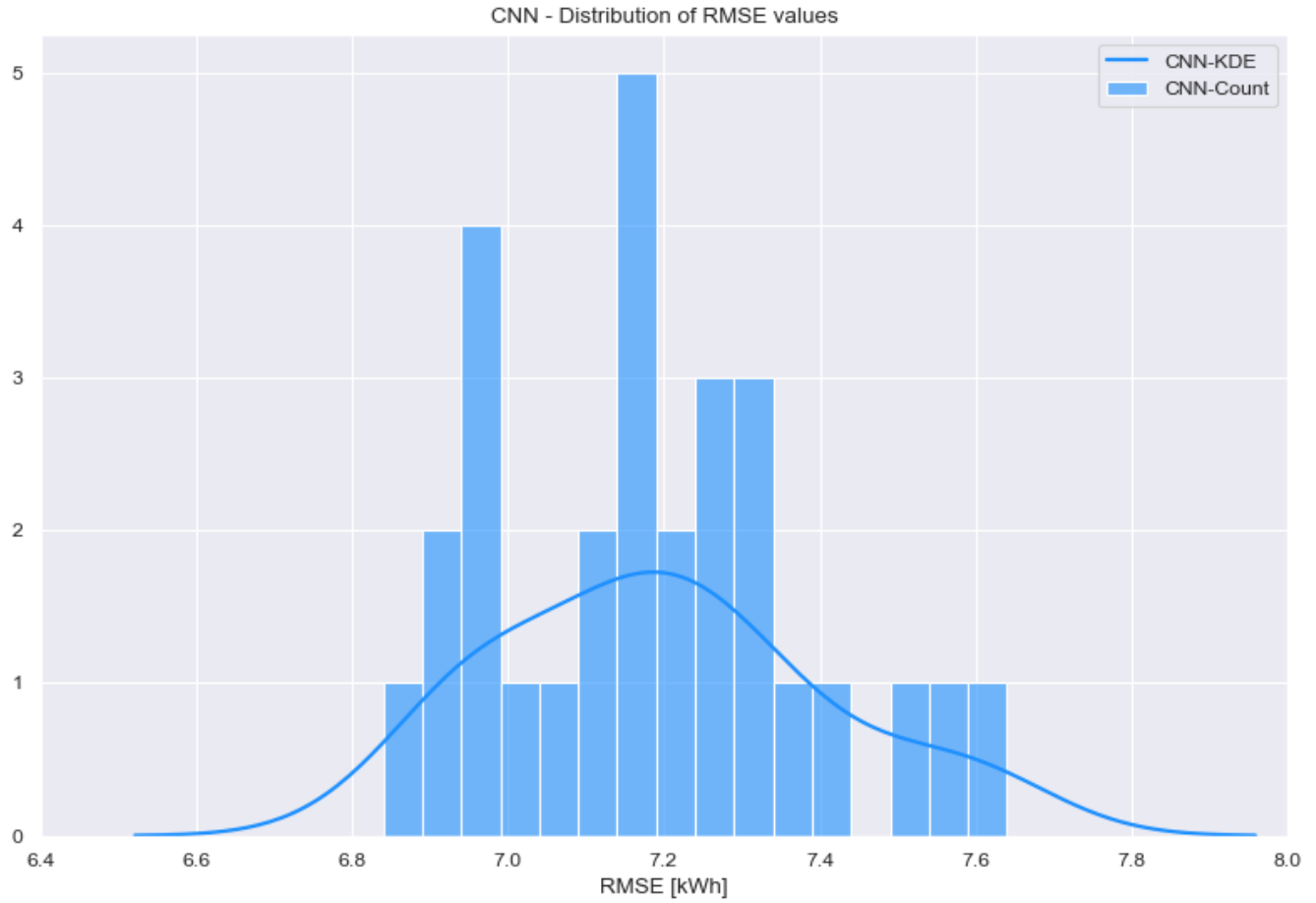


Figure 6.7: Distribution of RMSE values for CNN.

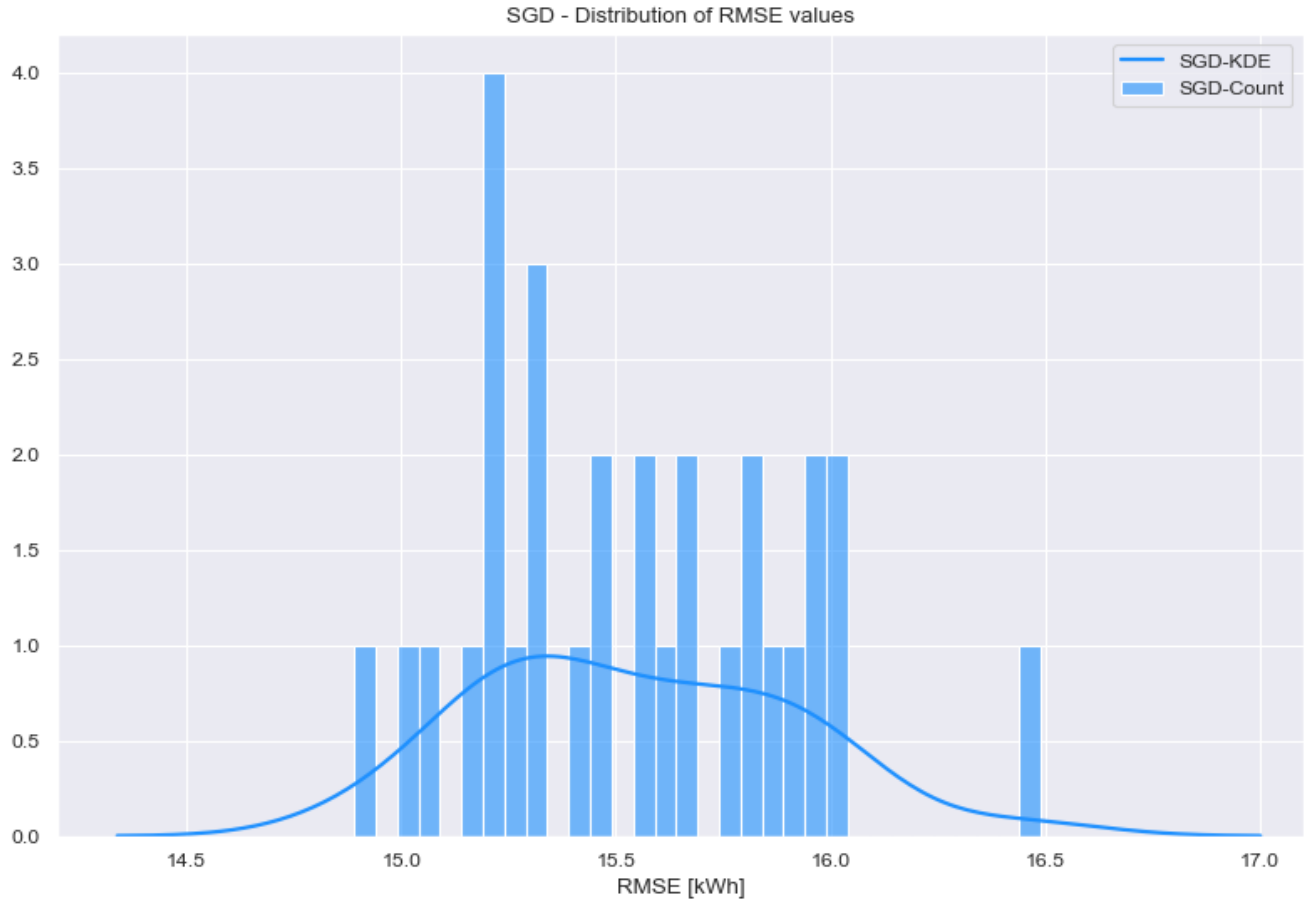


Figure 6.8: Distribution of RMSE values for SGD.

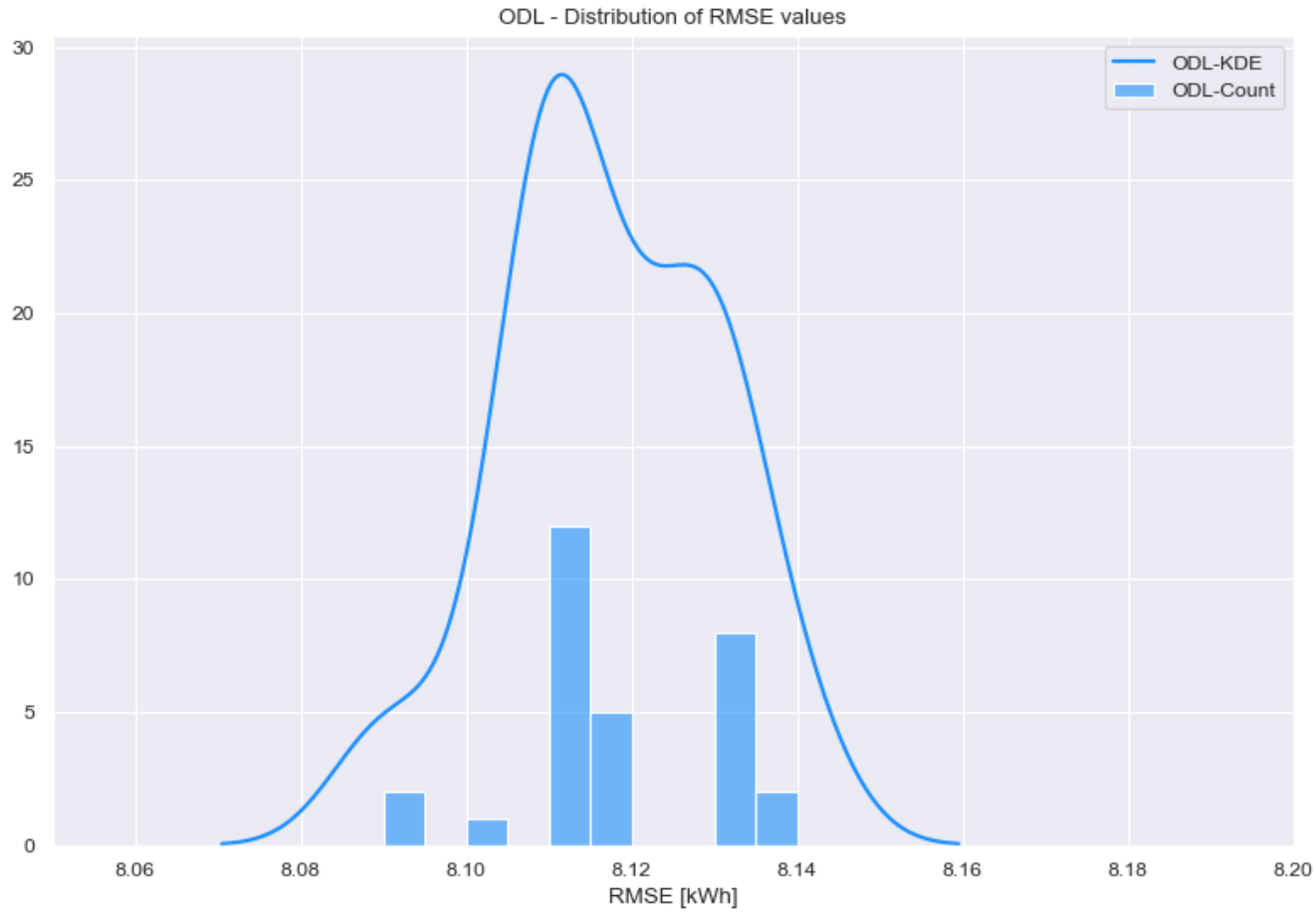


Figure 6.9: Distribution of RMSE values for ODL.

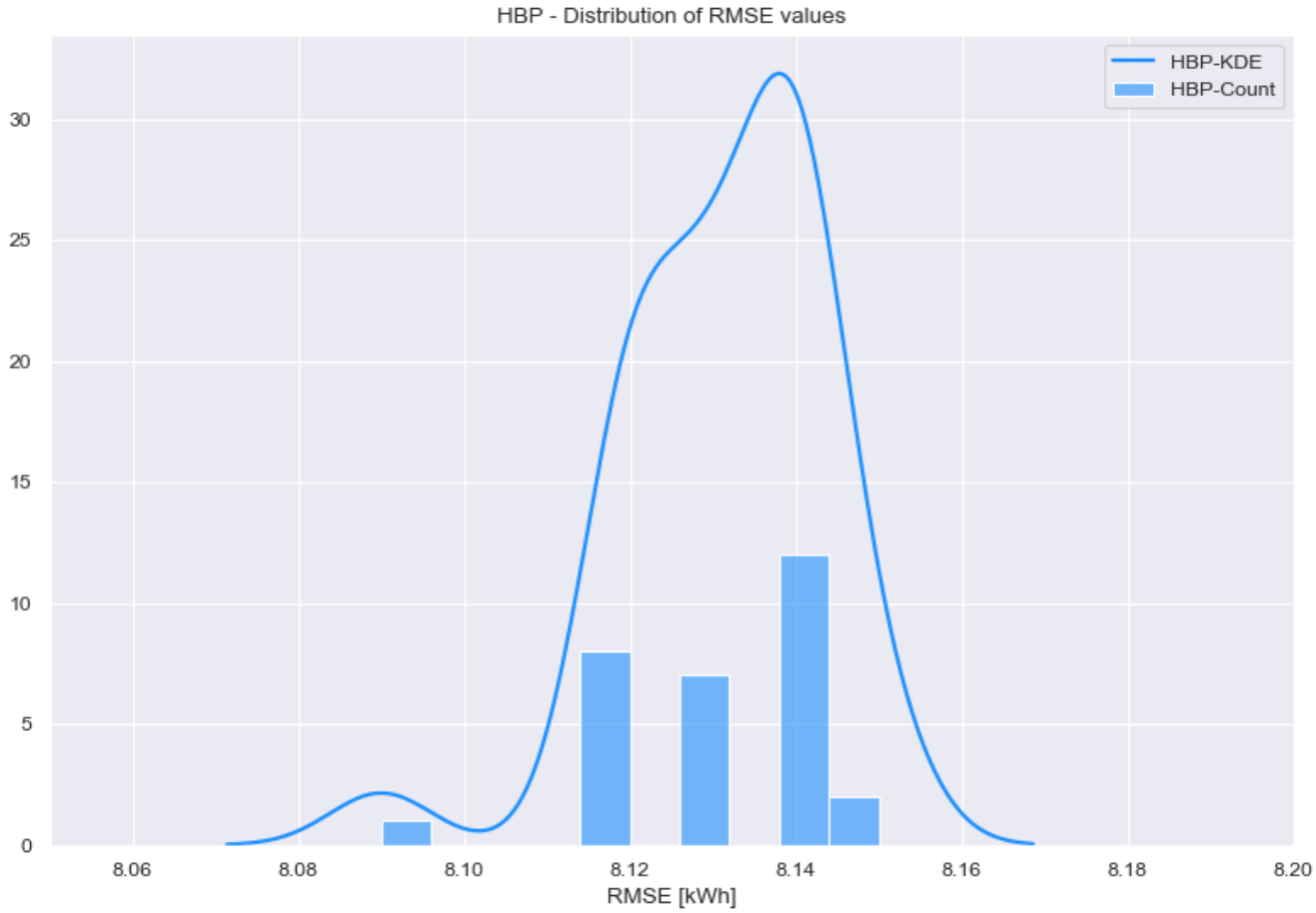
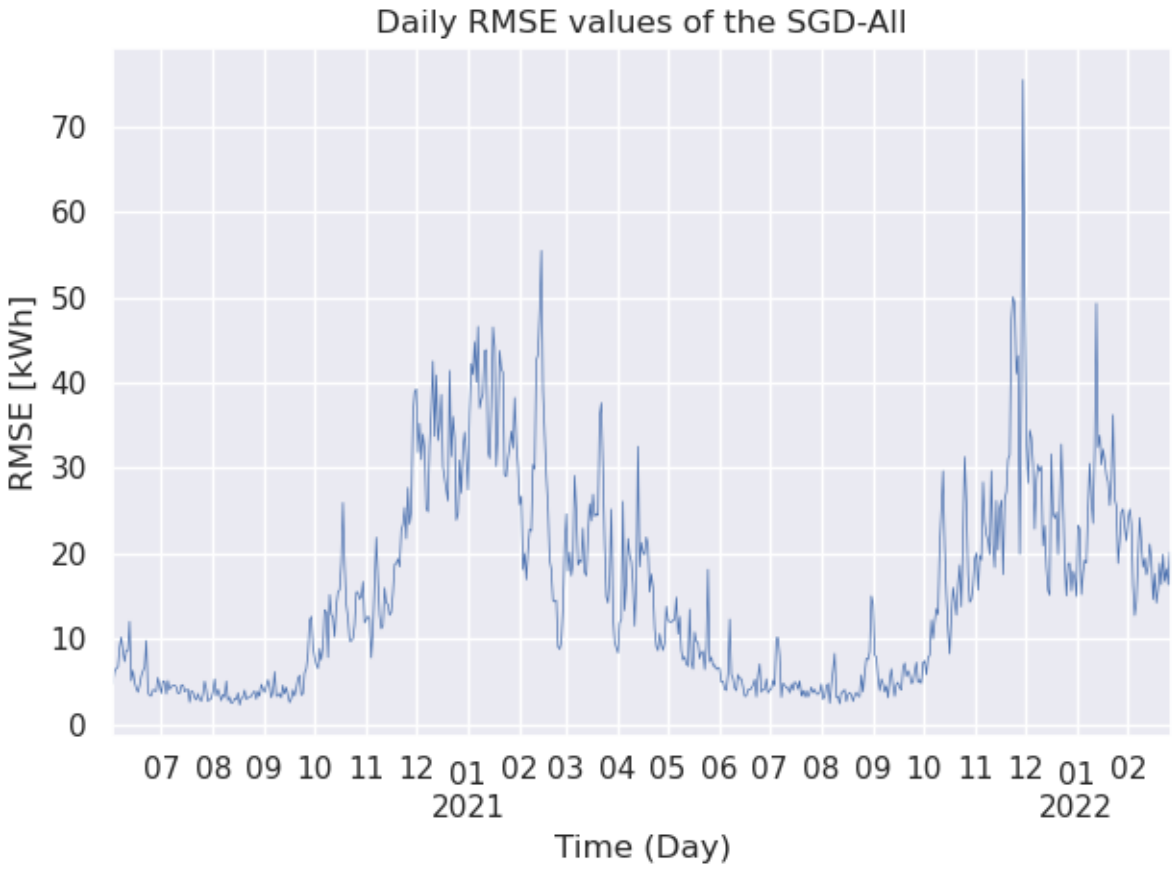
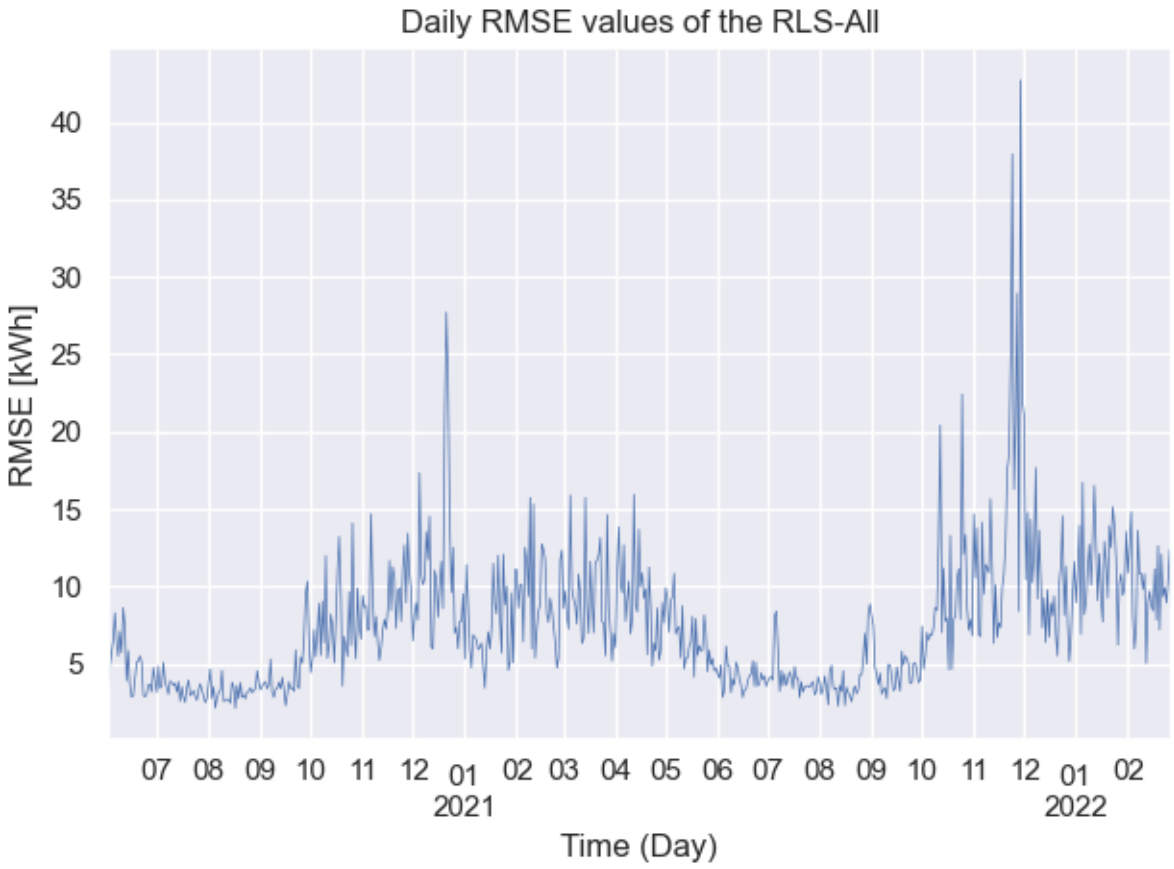


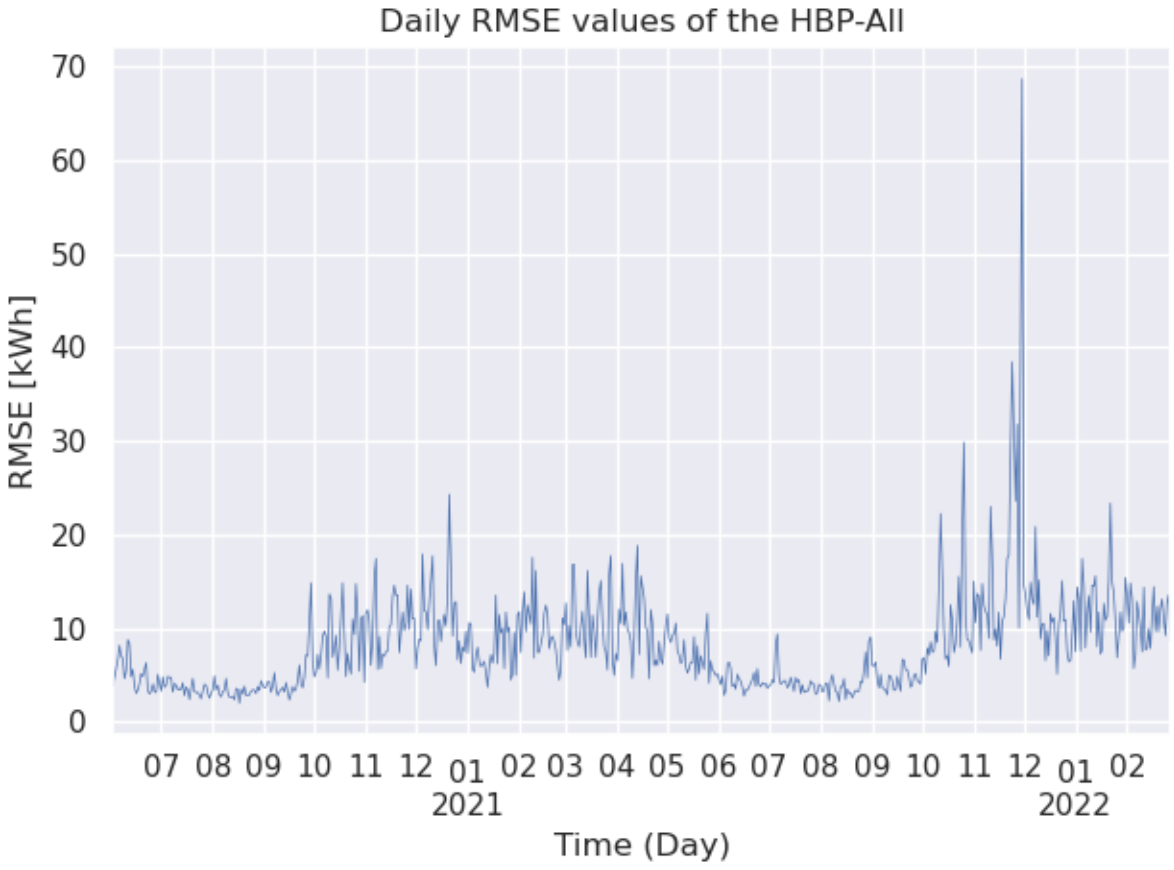
Figure 6.10: Distribution of RMSE values for HBP.

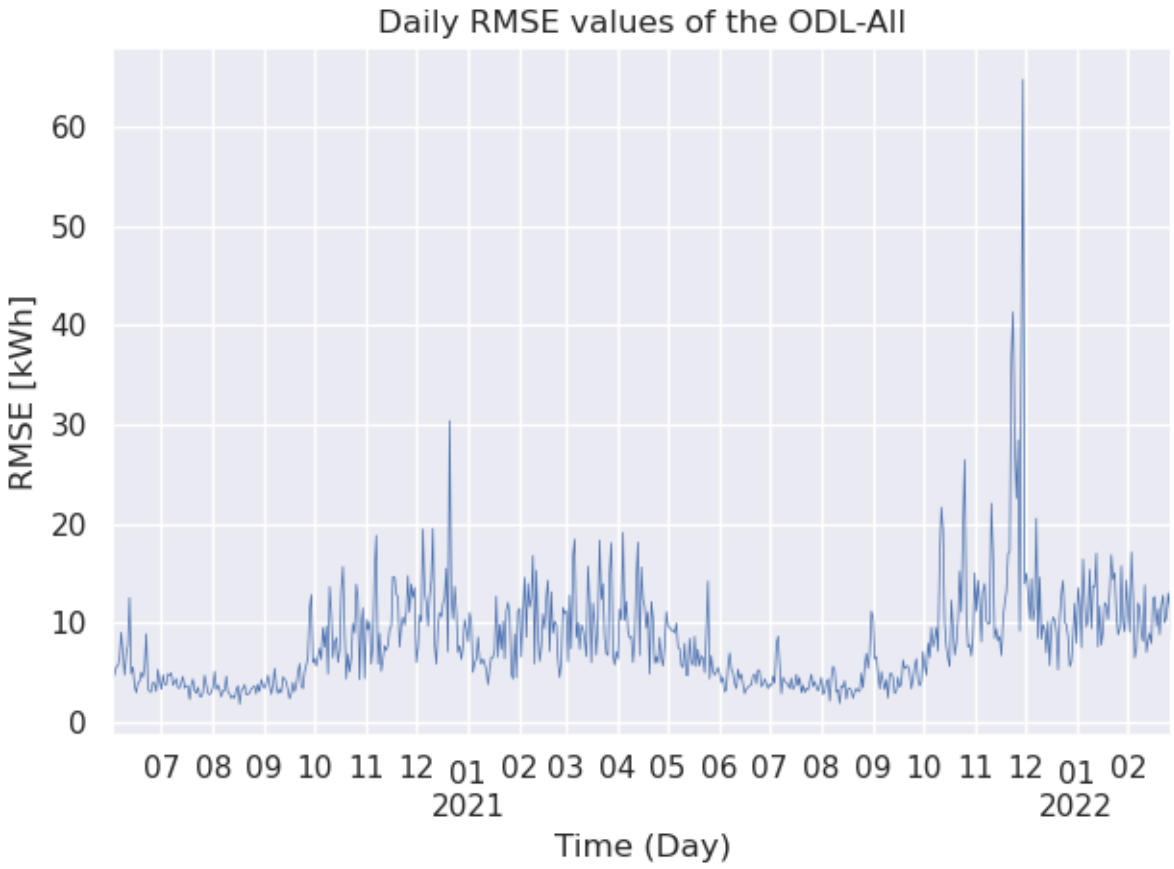
Section 7: The time course of the RMSE values for the online machine learning models.

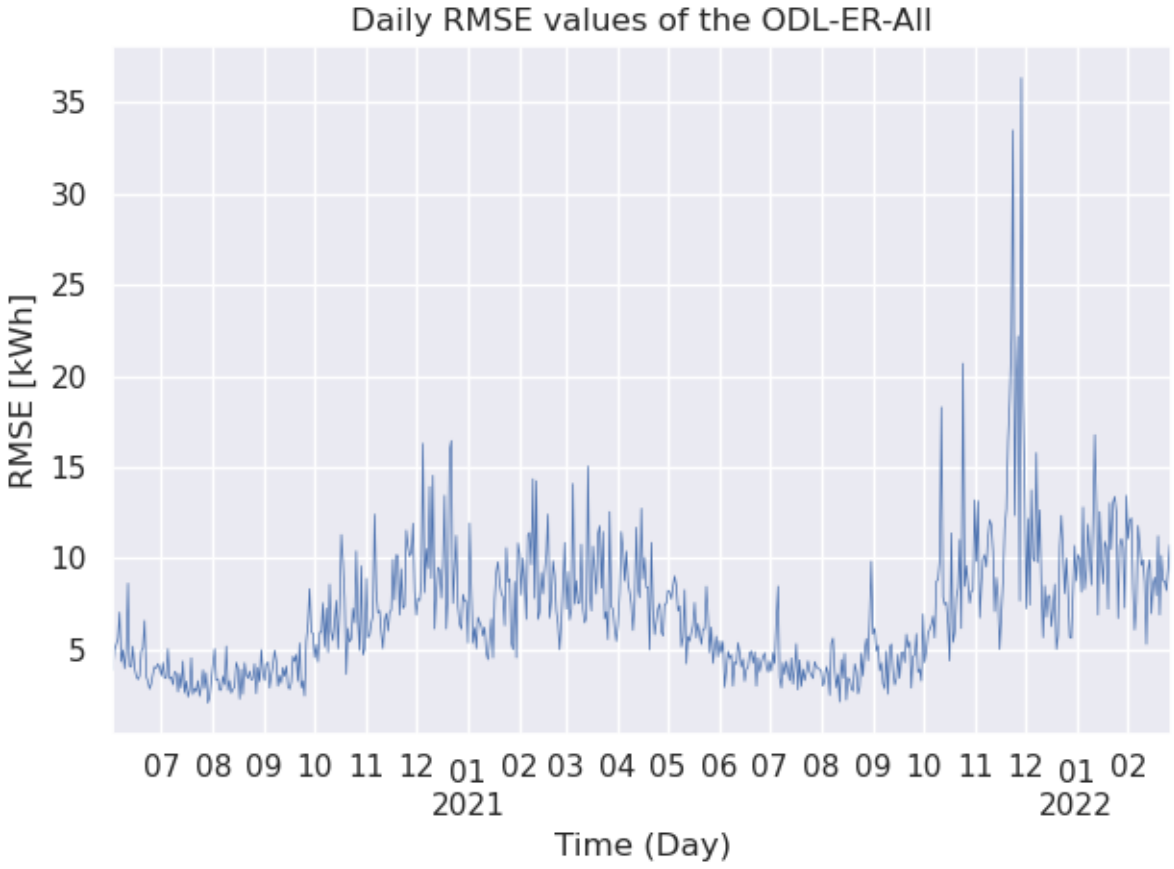
The figures show the progression of RMSE values for the five online machine learning models (Stochastic Gradient Descent (SGD), Recursive Least Squares (RLS), Online Deep Learning (ODL), ODL with an Experience Replay (ODL-ER) and Hedge Backpropagation (HBP)) trained and evaluated on the whole dataset.




Daily RMSE values of the RLS-All
RMSE [kWh]
40
35
30
25
20
15
10
5
07 08 09 10 11 12 01 02 03 04 05 06 07 08 09 10 11 12 01 02
2021
2022
Time (Day)


Daily RMSE values of the HBP-All
RMSE [kWh]
70
60
50
40
30
20
10
0
07 08 09 10 11 12 01 02 03 04 05 06 07 08 09 10 11 12 01 02
2021
2022
Time (Day)


Daily RMSE values of the ODL-All
RMSE [kWh]
0
10
20
30
40
50
60
07 08 09 10 11 12 01 02 03 04 05 06 07 08 09 10 11 12 01 02
2021
2022
Time (Day)


Daily RMSE values of the ODL-ER-All
RMSE [kWh]
5
10
15
20
25
30
35
07 08 09 10 11 12 01 02 03 04 05 06 07 08 09 10 11 12 01 02
2021
2022
Time (Day)