Renewable Energy 89 (2016) 671–679

Contents lists available at ScienceDirect

Renewable Energy

journal homepage: www.elsevier.com/locate/renene

Machine learning ensembles for wind power prediction☆

Justin Heinermann*, Oliver Kramer

Department of Computing Science, University of Oldenburg, 26111 Oldenburg, Germany

ARTICLE INFO

Article history:
Received 6 April 2015
Received in revised form
7 November 2015
Accepted 28 November 2015
Available online 29 December 2015

Keywords:
Wind power prediction
Machine learning ensembles
Multi-inducer
Heterogeneous ensembles
Decision trees
Support vector regression

ABSTRACT

For a sustainable integration of wind power into the electricity grid, a precise prediction method is required. In this work, we investigate the use of machine learning ensembles for wind power prediction. We first analyze homogeneous ensemble regressors that make use of a single base algorithm and compare decision trees to k-nearest neighbors and support vector regression. As next step, we construct heterogeneous ensembles that make use of multiple base algorithms and benefit from a gain of diversity among the weak predictors. In the experimental evaluation, we show that a combination of decision trees and support vector regression outperforms state-of-the-art predictors (improvements of up to 37% compared to support vector regression) as well as homogeneous ensembles while requiring a shorter runtime (speed-ups from 1.60× to 8.78×). Furthermore, we show the heterogeneous ensemble prediction can be improved when using high-dimensional patterns by increasing the number of past steps considered and hereby the spatio-temporal information available by the measurements of the nearby turbines. The experiments are based on a large wind time series data set from simulations and real measurements.

1. Introduction

For a successful integration of wind energy into the power grid, precise forecasts are needed. Besides numerical weather predictions, machine learning algorithms yield good forecasting results [1,2]. A recent comparison shows that machine learning predictors are well suited to short-term predictions with forecast horizons up to a few hours [3]. Especially when spatio-temporal information is available, *k-nearest neighbors* (*k*-NN) and *support vector regression* (SVR) can give feasible prediction performance. Fig. 1 shows the wind speed measurements for a set of turbines near Reno: It can bee seen that nearby turbines show similar speeds at the same time and it exists some correlation between the time series of them. If one wants to give a power output prediction for a certain turbine based on its past time series measurements used as patterns, including the measurements of turbines in the vicinity of a few kilometers into the patterns greatly helps to reduce the prediction error. However, there are two big challenges when applying machine learning techniques to the field of wind power prediction: First, in order to reach the best prediction accuracy possible with these algorithms, the computation time grows very large. E.g., training an SVR can easily take hours – strongly increasing with the number of considered neighboring turbines and past measurements. Second, the prediction performance needs to be improved further to cope with the actual energy markets needs. These two aspects make the choice of machine learning algorithm and parameters for wind power forecasting difficult.

It has been shown that a good alternative to the well-known machine learning algorithms is combining several basic models to *ensemble predictors*: By employing a number of so-called weak predictors and eventually combining their outputs to a prediction, the accuracy of classification and regression can be improved while reducing the computation time. Especially for real-world problems, machine learning ensembles are promising approaches. An example for ensemble regression applied to wind power prediction can be seen in Fig. 2. In contrast to state-of-the-art machine learning algorithms, ensemble methods require less tuning and expert domain knowledge. Nevertheless, in order to find an optimal ensemble predictor, usually a trade-off between multiple objectives has to be made: For example, a lower prediction error is often achieved by investing more computation time.

In this work, we discuss the practical use of *regression ensembles* for the task of wind power forecasting, aiming at optimal regression

☆ This paper was intended as part of the special issue entitled "Optimization Methods in Renewable Energy Systems Design" Volume 87, Part 2, Pages 835-1030 (March 2016). However, the article was accepted after the deadline for acceptance for this issue and we were unable to delay compilation of the issue. We apologize for any inconvenience this may cause.

* Corresponding author.
E-mail address: justin.heinermann@uni-oldenburg.de (J. Heinermann).

http://dx.doi.org/10.1016/j.renene.2015.11.073
0960-1481/

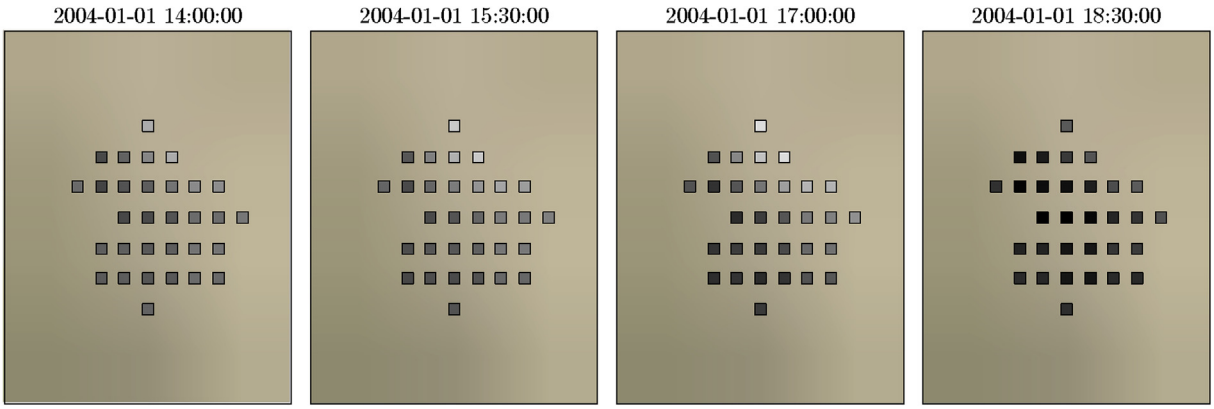


Fig. 1. Wind speed measurements for a wind park near Reno. The color denotes the wind speed, showing similar behavior of nearby turbines. A white color denotes no wind speed, a black color denotes a wind speed of nearly 30 m/s. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

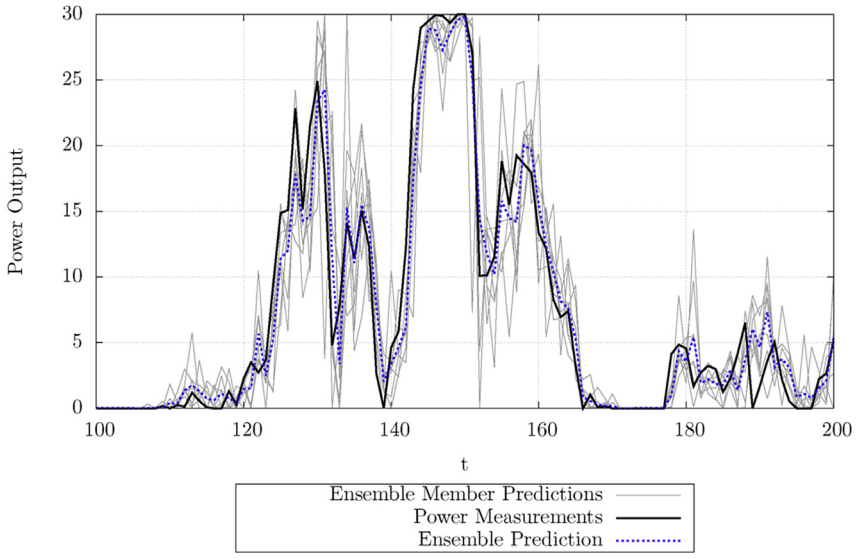


Fig. 2. Example for ensemble prediction of a wind power time series. While the eight single predictors do not provide a feasible prediction, their average gives a very good approximation to the real measurements.

accuracy as well as maintaining a reasonable computation time. In the first step we compare homogeneous ensemble predictors consisting of either *decision trees* (DT), *k*-NN, or support vector regressors as base algorithms. As diversity among the ensemble members is crucial for the accuracy of the ensemble, we propose the use of *heterogeneous ensemble predictors* consisting of different types of base predictors for wind power prediction. Our comprehensive experimental results show that a combination of DT and SVR yields better results than the analyzed homogeneous predictors while offering a decent runtime behavior. Going further, we show that heterogeneous ensemble predictors are very well-suited for using large numbers of neighboring turbines and past measurements and improve the prediction performance.

In the recent years, a strong increase in wind energy can be observed. E.g., there is an installed wind energy capacity of 128.8 GW in the EU, 11,791.4 MW of which has been installed in2014.[1] With growing capacity, there is also a growing demand of reliable and precise forecasts. The trading of wind energy at the power markets is only possible with good forecasts. At the electricity markets, there is a trend towards shorter forecast horizons. For instance, the lead time has been reduced to 30 min on all intraday markets of the EPEX Spot.[2] Other applications for short-term power predictions are planning of control energy, ensuring power grid stability, planning and charging of storage systems, and the planning of wind parks.

This paper is structured as follows: Section 2 gives an introduction to wind power prediction with machine learning in general, machine learning ensemble techniques, and an overview of related work. The application of regression ensembles to the field of wind power prediction and a comprehensive experimental analysis are presented in Section 3, followed by Section 4 dealing with heterogeneous ensemble predictors. The analysis of the question, how to use the largest possible amount of valuable information from the available data is done in Section 5. In Section 6, the heterogeneous ensemble prediction model is applied to the power output prediction of wind parks. Conclusions are drawn in Section 7.

[1] http://www.ewea.org/statistics/.

[2] https://www.epexspot.com/en/press-media/press/details/press/EPEX_SPOT_and_ECC_successfully_reduce_lead_time_on_all_intraday_markets.

2. Background

2.1. Machine learning approach to wind power prediction

We treat short term wind power prediction as a regression problem. In contrast to numerical weather predictions, machine learning methods usually make only use of the time series data itself. The history of measurements used to train the prediction model is called training data set $\mathbf{X} = \{(\mathbf{x}_1, y_1)\ldots,(\mathbf{x}_N, y_N)\} \subset \mathbb{R}^d \times \mathbb{R}$. When performing a forecast, the objective is to predict the expected value after a *forecast horizon* λ., e.g., in half an hour. The input patterns consist of a past of μ time steps, which we call the *feature window*. In this work, we use a spatio-temporal model based on the one proposed by Kramer, Gieseke, and Satzger [1], which showed the benefit of involving neighboring turbines to the input vector. Let $p_i(t)$ be the measurement of a turbine i at a time t, and $2 \leq i \leq (M+1)$ the indices of the M neighboring turbines. For a target turbine with index 1 we define a pattern-label-pair $(\mathbf{x},y)$ for a given time t_0 as

$$\begin{pmatrix} p_1(t_0-\mu) & \ldots & p_1(t_0) \\ \ldots & \ldots & \ldots \\ p_{(M+1)}(t_0-\mu) & \ldots & p_{(M+1)}(t_0) \end{pmatrix} \rightarrow p_1(t_0+\lambda) \quad (1)$$

In our experiments, we use the *NREL western wind resources dataset*.[3] It consists of simulated wind power output for 32,043 wind turbines in the US, given in 10-min time resolution for the years 2004–2006. For every turbine, there are 157,680 wind speed and power output measurements available. In our experiments, we use the power output data of five wind parks[4] that consist of the target wind turbine and the 10 neighbored turbines. The data from 01/2004 until 06/2005 is used as training data set and the data from 7/2005 until 12/2006 serves as test data set.

To measure the prediction accuracy, we employ the commonly used mean squared error (MSE). For a test set $\mathbf{X}_t = \{(\mathbf{x}'_1, y'_1)\ldots,(\mathbf{x}'_{N_t}, y'_{N_t})\} \subset \mathbb{R}^d \times \mathbb{R}$ and prediction model $f(\bullet)$, it is defined by:

$$E = \frac{1}{N_t}\sum_{i=1}^{N_t}\left(f(\mathbf{x}'_i) - y'_i\right)^2$$

The difference between the label y_i of a test instance and corresponding prediction $f(\mathbf{x}'_i)$ is squared to penalize bigger differences stronger. The mean of the squared prediction errors of the N_t test instances is then computed as overall accuracy measure.

2.2. Machine learning ensembles

The idea of ensemble methods can be described as building "a predictive model by integrating multiple models" [4]. One of the advantages is the possible improvement of prediction performance. Another reason for utilizing ensemble methods is the reduction of computational complexity, which can be helpful on very large data sets. There are countless variants of different ensemble algorithms. A comprehensive overview and empirical analysis for ensemble classification is given by Bauer and Kohavi [5]. Another, more up-to-date review paper is given by Rokach [4].

An important and famous approach is *Bagging*, which stands for bootstrap aggregating, and was introduced by Breiman [6]. The main idea is to build independent predictors using samples of the training set and average (or vote) the output of these predictors. Breiman shows that bagging ensembles of decision trees as well as regression trees work very well in comparison with single trees. Furthermore, he gives arguments for the question "why bagging works" [6]. A popular variant of bagging approaches is the random forest algorithm [7] that "uses a large number of individual, unpruned decision trees" [4]. Every decision tree is built with a subset sample from the training set, but only uses N of the available features of the patterns.

There exist more sophisticated ensemble approaches like AdaBoost [8] or *Stacked Generalization* [9], but, as we want to give a proof of concept with the possibility of heterogeneity, we limit ourselves here to bagging.

One key ingredient to successful building of ensembles is the concept of diversity: All the weak predictors should behave different if not uncorrelated [4,10]. Then, the ensemble prediction improves. There are many ways to generate such diversity, like manipulating the used training sample, the used features, and the weak predictors' parameters. Another possibility is the hybridization of multiple algorithms, which we call heterogeneous ensembles.

2.3. Related work

It has been shown that machine learning methods are well-suited to the domain of wind speed and wind power prediction. Techniques like k-NN [11,2] or neural networks [12] have successfully been applied. In a recent comparison, Treiber et al. [3] show that support vector regression is superior to numerical weather predictions for shortest-term forecast horizons. Up to 3 h, machine learning techniques are superior to post-processed meteorological models. For more than 6 h, meteorological methods should be preferred. Since short-term forecasts have applications as well, we want to improve predictions for the short-term time scale. In the future, a hybridization with meteorological methods could be beneficial for different forecast horizons.

The application of machine learning ensembles to the field of wind power prediction has also been shown to work well: Kusiak, Zhang and Song [13] successfully apply different methods to short-term wind power prediction, one of which is the bagging trees algorithm. Fugon et al. [14] compare various algorithms for wind power forecasting and show that random forests with and without random input selection yield a prediction performance similar to SVR, but recommend to prefer a linear model when the computation time grows too large. Heinermann and Kramer [15] achieve good wind power prediction results using support vector regression ensembles. Another similar application of ensembles is given by Hassan, Khosravi, and Jaffar [16] for electricity demand forecasting. Here, neural network ensembles are applied. A key for the success is, again, the diversity amongst the predictors. For solar power output prediction, Chakraborty et al. [17] built up an ensemble of a weather forecast-driven Naïve Bayes Predictor as well as a kNN-based and a Motif-based machine learning predictor. The results of the three predictors are combined with a Bayesian Model Averaging. Chakraborty et al. show that the prediction error can be reduced by inducing ensemble methods to forecasting power output.

Besides numerical weather forecasts and statistical methods, models based on computational fluid dynamics (CFD) gained attention in the recent past. Marti et al. [18] proposed a model feasible for forecast horizons up to 4 h. Castellani et al. [19] investigate the hybridization of CFD and artificial neural networks.

In the field of numerical weather forecasts, it is quite common to use ensemble postprocessing: Gneiting et al. [20] found ensembles to reduce the prediction error by applying the ensemble model

[3] http://wind.nrel.gov/.

[4] The IDs of the turbines in the NREL dataset are: Cheyenne = 17423, Lancaster = 2473, Palmsprings = 1175, Vantage = 28981, Yucca Valley = 1539.

output statistics (EMOS) method to diverse weather forecasts. Similarly, Thorarinsdottir and Gneiting [21] are using a so-called "heteroscedastic censored regression" for maximum wind speed prediction over the American Pacific Northwest.

3. Regression ensembles for wind power prediction

Our objective is to find out if heterogeneous machine learning ensembles are a superior alternative to state-of-the-art machine learning predictors. We decided to implement a relatively simple bagging approach with weighting, which has some advantages. While the implementation is straight-forward and offers a moderate computational cost, we consider the approach sufficient for a proof of concept, which is also shown in the experimental evaluation. Another good example for this ensemble algorithm is the famous Random Forest method that yields very good results, too, and is relatively fast compared to Boosting algorithms. The latter ones could outperform for some applications, but also tend to overfitting, and can hardly be parallelized. A comparison to more sophisticated ensemble approaches like AdaBoost as well as stacked generalization will be subject to future work.

3.1. Algorithmic framework

Our algorithm is outlined in Algorithm 1. As usual in supervised learning, a training set $\mathbf{X}$ with known labels is given. The most important meta-parameters of the algorithm are the number n of weak predictors, the number s of samples, and the types of base algorithms used for each predictor. Both n and s have to be chosen large enough in order to provide satisfying results. However, the best choice for n and s depends on the base algorithm used, which also influences the runtime significantly. The main work of the algorithm takes place in the for-loop beginning in line 3. Each pass trains one weak predictor p_i and its assigned weight w_i. For each weak predictor, a subset of $\mathbf{X}$ with size s is sampled and used as training set T_i for the particular predictor p_i. The sampling can be done with or without replacement, which we will discuss in the experimental part. In order to calculate weight w_i, the remaining training patterns are used as a validation set T_{val}. The value w_i is then obtained by testing p_i on T_{val} and taking the inverse of the mean squared error (MSE).

Algorithm 1 Training of Ensemble Predictor

1: **Inputs:**
   $\mathbf{X} = \{(\mathbf{x}_1, y_1) \ldots, (\mathbf{x}_N, y_N)\} \subset \mathbb{R}^d \times \mathbb{R}$
   Number of predictors: n
   Number of samples: s
   Algorithms to use: $\mathbf{A} = \{a_i | i \in 1 \ldots n\}$
2: **Returns:**
   Predictors: $\mathbf{P} = \{p_i | i = 1, \ldots, n\}$
   Weights: $\mathbf{W} = \{w_i \in \mathbb{R} | i = 1, \ldots, n\}$
3: **for** i = 1 **to** n **do**
4: $\quad \mathbf{X}_{sample} \leftarrow sample(\mathbf{X}, s)$
5: $\quad \mathbf{X}_{val} \leftarrow \mathbf{X} - \mathbf{X}_{sample}$
6: $\quad p_i \leftarrow trainPredictor(a_i, \mathbf{X}_{sample})$
7: $\quad w_i \leftarrow \frac{1}{MSE(p_i, \mathbf{X}_{val})}$
8: **end for**

When the training algorithm computed the predictors and weights, for an unknown instance $\mathbf{x}'$ the predicted label is computed by:

$$f(\mathbf{x}') = \frac{\sum_{i=1}^{k} w_i \cdot p_i(\mathbf{x}')}{\sum_{i=1}^{k} w_i} \tag{2}$$

Each predictors output $p_i(\mathbf{x})$ is computed and then weighted by w_i in the resulting weighted average. In a realistic scenario, one would perform all calls of p_i in parallel using multi- or manycore processors. Because there are no computational dependencies between the ensemble members, the problem is embarrassingly parallel. To give an easy and fair comparison, in our experiments we only employ only one CPU core for the runtime measurements. As depicted in Table 1, the runtime for training depends on the base algorithm used, the number of estimators employed, and the number of samples used. E.g., an ensemble of 32×500 DT ensemble regressor can be trained in only 1 s, whereas a $256 \times 1{,}000$ SVR ensemble needs more than 10 min. Because the different ensemble predictors yield different prediction accuracies, the computation time should be considered when choosing a model for practical use. In Section 4, we investigate the problem of giving the best possible prediction in a short computation time.

As pointed out by Ho [22], random forests and bagging classifiers in general might benefit from sampling from the feature space, i.e., taking only a random subset of the available features into account. Besides the number of features used, the choice can be done with or without replacement. Although replacement is sometimes considered as useful [4,23], there is no explicit rule when to use it. In a preliminary experiment, we found no evidence that random feature subspaces help to increase the accuracy, but of course can help to decrease runtime. Because we could not find a significant difference between sampling with replacement and sampling without replacement, we employ sampling without replacement. For the basic comparison of the weak predictors, all features are used. For the comparison of heterogeneous ensembles with state-of-the-art predictors, the number of features sampled will be considered again because of the possible trade-off between runtime and accuracy. Concerning the sampling from the training set, we also found no evidence for the supremacy of replacement, but due to the recommendations in literature [6,4], we decided to employ sampling with replacement in the following experiments.

3.2. Choice of the base algorithms and training of weak predictors

Since the number of possible settings is huge, one has to make some assumptions to limit the number of combinations. First, we want to give an overview over the choice of base algorithms and parameters.

The decision tree algorithm is a powerful yet simple tool for supervised learning problems [24]. The main idea is to build a binary tree, which partitions the feature space into a set of rectangles. The assignment of the unknown pattern to one of these rectangles is used for computing the sought label. While there are different algorithms for building up decision trees, we limit ourselves to the famous CART algorithm [25]. Besides moderate computational costs, the main advantage of decision trees is their interpretability.

The SVR algorithm often provides very good prediction results and is regarded as state-of-the-art regression technique. In general, the support vector machine (SVM) algorithm maximizes a geometric margin between the instances of the classes to be separated. Similar to SVM classification, the SVR algorithm aims at finding a prediction function $\widehat{f} : \mathbb{R}^d \times \mathbb{R} \rightarrow \mathbb{R}$ that computes an accurate prediction value for an unseen pattern $\mathbf{x} \in \mathbb{R}^d$. For more detailed information about the algorithm, we refer to, e.g., [26]. We utilized a RBF kernel and choose $C = 10{,}000$ and $\sigma = 1e{-}5$ for this experiment. In the following sections, cross-validation is utilized for parametertuning.

A famous yet relatively simple approach for classification and regression is the *k-nearest neighbors* (k-NN) model, see Ref. [27]. The prediction averages the label information of the k nearest neighbors, i.e., via $f(\mathbf{x}) = 1/k \sum_{i \in N_k(\mathbf{x})} y_i$, where N_k denotes the set of

Table 1
Comparison (MSE) of ensemble predictors consisting of different base algorithms used as weak predictor. For every turbine, the best result is printed in bold. Each experiment has been repeated 10 times.

Base algorithm	DT		SVR						*k*-NN			
n	32	32	256	256	32	32	256	256	32	32	256	256
s	500	1000	500	1000	500	1000	500	1000	500	1000	500	1000
Casper	11.17	10.93	10.89	**10.62**	10.99	10.84	10.95	10.87	13.10	12.44	13.02	12.44
Las Vegas	10.84	10.61	10.51	10.27	**10.26**	**10.26**	10.27	10.27	12.84	12.36	12.81	12.30
Hesperia	7.98	7.82	7.76	**7.59**	7.62	7.61	7.60	**7.59**	9.41	8.96	9.36	8.98
Reno	14.76	14.53	14.47	14.19	14.11	14.10	14.00	**13.98**	18.92	18.03	19.14	18.14
Vantage	7.31	6.97	7.00	6.83	6.61	6.58	**6.57**	**6.57**	8.44	7.93	8.43	8.07
Approx. t_{train} (s)	1	2	10	15	60	120	600	1200	36	60	292	481

indices for the k nearest neighbors in T w.r.t. a distance metric like Euclidean distance. While a naïve implementation takes $\mathcal{O}(|S| \cdot |T|)$ time for a training set T and a test set S, more efficient implementations with spatial data structures, e.g. k-d trees, are available [28,24]. Logarithmic runtime is offered for small dimensionalities $d \leq 15$. For parameter k, we make a random choice in the interval [1;25].

We experimentally compare different regression algorithms composed to ensembles: Table 1 shows a comparison of decision trees, SVR, and k-NN used as weak predictors. A general observation is that increasing n and s decreases the prediction error. With the given n and s, no clear decision between decision trees and SVR can be made, but we stick to these two basic algorithms in the further experiments rather than k-NN.

4. Heterogeneous ensembles

Of course, when dealing with forecasting tasks, the first goal is to reach the lowest prediction error possible. While it is possible to decrease the prediction error by using ensembles, a feasible runtime is equally important for practical relevance. If it takes hours to train a regressor for one turbine, a model would be unusable for a large number of turbines requiring a forecast – the time needed for parameter-tuning and cross-validation not mentioned. Therefore, our goal is to reach a good prediction performance as well as a short runtime for both training and testing. As seen in Table 1, there is no clear answer which algorithm should be preferred. Since it is known that ensemble predictors benefit from diversity amongst the ensemble members, it a heterogeneous choice of base algorithms could yield a better regression accuracy as well. In this work, we propose to use heterogeneous ensembles built upon SVR and decision tree regressors. We chose the most simple approach by training one SVR ensemble and one DT ensemble. The predicted value is then obtained by computing the mean the two ensembles' predictions. As shown in the following experiments, the result is a robust prediction algorithm that offers a reasonable runtime behavior. The experiments in the following section address the question, if heterogeneous ensembles offer a better performance than homogeneous ensembles. The second question is: Can heterogeneous ensembles help to decrease the computation time needed?

In our experiments, we analyze heterogeneous ensembles built upon SVR and decision tree regressors. In our experiments, we use the power output data of the five wind parks and include the measurements of 10 neighbored turbines into the patterns. As training data set, the data from 01/2004 until 06/2005 is used and the data from 7/2005 until 12/2006 serves as test data set. The experiments were run on an Intel Core i5 at 3.10 GHz with 8GB of RAM. The algorithms were implemented in Python utilizing the *k*NN, decision tree, and SVR implementations of *Scikit-learn* [29].

One has to consider that different weak predictors show different behavior and could benefit from different combinations of n and s. I.e., we will see that a large amount of predictors is a good possibility to ameliorate decision tree ensembles while the runtime does not suffer as much as in SVR ensembles. Instead of combining some SVR predictors and some decision trees, one could possibly better combine a huge number of decision trees and maybe also increase sample number s.

Therefore, we have to give a fair comparison, which considers both prediction performance and runtime. First, we analyze the behavior of the homogeneous ensembles based on SVR or decision trees. We try to find good combinations, which are computable in a feasible time. The result can be seen in Table 2: Like assumed, one can train more decision trees with a larger sample in the same time as SVR predictors. The central point of the experiment is the equally-weighted combination of one SVR ensemble and one DT ensemble at a time to one heterogeneous ensemble.

Table 2
Behavior of different setups of SVR, DT, and combined ensembles for a wind turbine near Las Vegas.

Setup	n	s	t_{train}	t_{test}	MSE
(a) SVR ensembles					
1	4	500	7.58	7.11	10.78
2	4	1000	14.69	13.09	10.43
3	4	2000	29.94	25.46	10.58
4	32	500	59.73	55.91	10.30
5	32	1000	117.86	105.61	10.26
6	32	2000	245.85	206.68	10.35
7	64	500	119.40	111.91	10.20
8	64	1000	236.26	211.83	10.26
9	64	2000	489.92	410.81	10.26
(b) Decision tree ensembles					
1	32	2000	4.91	3.65	10.44
2	32	4000	7.74	3.85	10.27
3	32	40,000	81.96	3.82	10.06
4	256	2000	42.67	28.86	10.13
5	256	4000	63.17	29.17	10.05
6	256	40,000	733.15	30.35	9.86
7	1024	2000	161.60	113.97	10.15
8	1024	4000	258.03	115.55	10.03
9	1024	40,000	2859.18	136.71	9.82

(c) Combinations of one DT ensemble and on SVR ensemble to one heterogeneous ensemble. For every combination, the MSE is shown. The row denotes which SVR ensemble is employed, whereas the column shows which DT ensemble is used.

	D1	D2	D3	D4	D5	D6	D7	D8	D9
S1	10.13	10.04	9.87	10.05	9.99	9.82	10.06	9.99	9.81
S2	10.03	9.96	9.75	9.97	9.91	9.70	9.97	9.90	9.69
S3	10.05	9.99	9.81	9.99	9.94	9.77	10.00	9.94	9.76
S4	10.02	9.95	9.76	9.95	9.90	9.72	9.97	9.89	9.71
S5	9.96	9.89	9.72	9.90	9.85	9.67	9.91	9.84	9.66
S6	9.98	9.92	9.75	9.92	9.87	9.70	9.93	9.86	9.70
S7	9.98	9.91	9.72	9.91	9.86	9.68	9.92	9.86	9.67
S8	9.99	9.92	9.74	9.92	9.87	9.69	9.93	9.86	9.68
S9	9.97	9.90	9.73	9.90	9.85	9.68	9.91	9.84	9.67

The results of these combinations are depicted in Table 2(c), which has the form of a matrix. In every cell of the matrix, the used SVR ensemble is given by the row and the used decision tree ensemble is given by the column. In the table, only the MSE is given for clear arrangement. The training and test times for one predictor is approximately the sum of the respective times of the two combined ensembles. Besides the very promising results, which outperform the homogeneous ensembles, we also can see that the combination of two weaker ensembles takes less time to deliver the same prediction error. We visualize this behavior for two wind turbines in Fig. 3.

In Table 3, we give a comparison of our heterogeneous ensemble method to SVR and k-NN which are considered as state-of-the-art regressors. The parameters k and accordingly C and σ were optimized with a 10-fold cross-validation. The training times are measured using the optimal parameters, so the huge amount for parameter tuning is not included. The testing times showed similar behavior. As comparison, we show two different heterogeneous ensembles. Both consist of one SVR ensemble with $n = 32$ an $s = 1{,}000$ and one decision tree ensemble with $n = 256$ and $s = 10{,}000$. The first one uses all 33 features available, whereas the second only makes use of 15 randomly chosen features without replacement. The result of this comparison is that in most cases the ensemble predictor outperforms classical SVR. Further, the training time is much shorter. If one must make a trade-off and decrease training or testing time, he might want to use a feature-reduced variant.

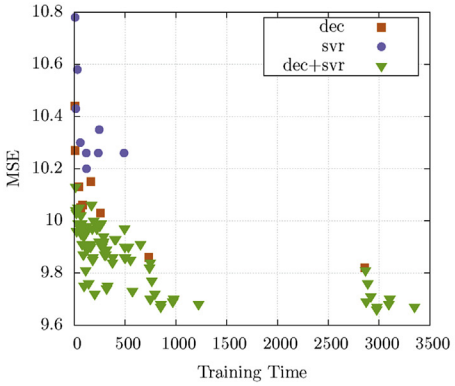
Table 3
Comparison of MSE and training time for five wind turbines. Our ensemble using all features yields the best MSE in four cases, but only takes a small amount of the time taken by standard SVR. If training time is considered more important than MSE, on can reduce the number of features used without loosing much of prediction performance.

Turbine	k-NN	SVR	ENS33	ENS15
(a) Test error (MSE)				
Casper	10.67	9.88	10.19	10.40
Hesperia	7.69	7.39	7.25	7.44
Las Vegas	10.46	15.69	9.78	10.11
Reno	14.81	13.29	13.16	13.83
Vantage	6.86	6.54	6.41	6.65
(b) Training time (s)				
Casper	1	704	413	177
Las Vegas	1	1450	378	165
Hesperia	1	1218	387	173
Reno	2	1173	399	173
Vantage	2	601	374	165
(c) Test time (s)				
Casper	97	268	214	114
Las Vegas	118	341	206	108
Hesperia	83	261	227	113
Reno	98	253	205	113
Vantage	105	251	229	108

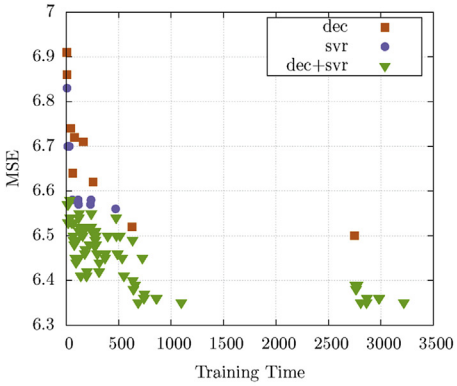
5. Increasing the number of used features

Besides the parameter-tuning of the algorithms, it must be evaluated how large the number M of considered neighbor turbines and number μ of past time steps should be to get the best prediction error. I.e., one wants to use as much valuable information from the data as possible. Like shown before, there is valuable information existent in the neighboring turbines and in the past time steps. However, with increasing number of features, the data become more and more challenging for the employed regression algorithms: First, the computational time is often dependant on the dimensionality of the data. Second, the prediction accuracy can get worse. E.g., when considering k-NN with Euclidean distance measure, the expressiveness of the distance between two instances is decreased with a higher dimensionality. The computation time also grows large, and for dimensionalities $d > 15$ it is hardly possible to benefit from spatial data structures like k-D-Trees [28]. For the practical use of our proposed heterogeneous ensemble predictors, we have to analyze the abilities of the algorithmic framework to make use of as much information in the data as possible and improve the prediction further with tuning of the number of considered neighbor turbines and the past measurements of these and the target turbine.

In a first experiment, we show the behavior of different regression algorithms when varying the number M of neighbor turbines and the feature window μ. While for k-NN and SVR regressors are sought by a grid search and a 3-fold cross validation, for the ensemble predictors we chose a fixed setting. Fig. 4 shows the MSE depending on M and μ for a turbine near Vantage. For k-NN and k-NN ensembles, the best prediction error is reached with small M and μ, i.e., a small number of features. We assume that k-NN ensembles perhaps could benefit from random feature subsets in order to deal with higher dimensionalities. The SVR predictor as well as the ensemble predictors show a different behavior: With a larger number of neighbor turbines included into the features, the prediction error gets smaller. But the feature window has to be increased as well to achieve better results. With a physical



(a) Las Vegas: MSE depending on training time



(b) Vantage: MSE depending on training time

Fig. 3. Behavior of runtime and prediction performance for homogeneous and heterogeneous ensembles for two wind turbines near Las Vegas and Vantage. The heterogeneous combinations can outperform the homogeneous ensembles. In particular, the solutions in each bottom left corner show ensembles with a very short computation time as well as a very low error.

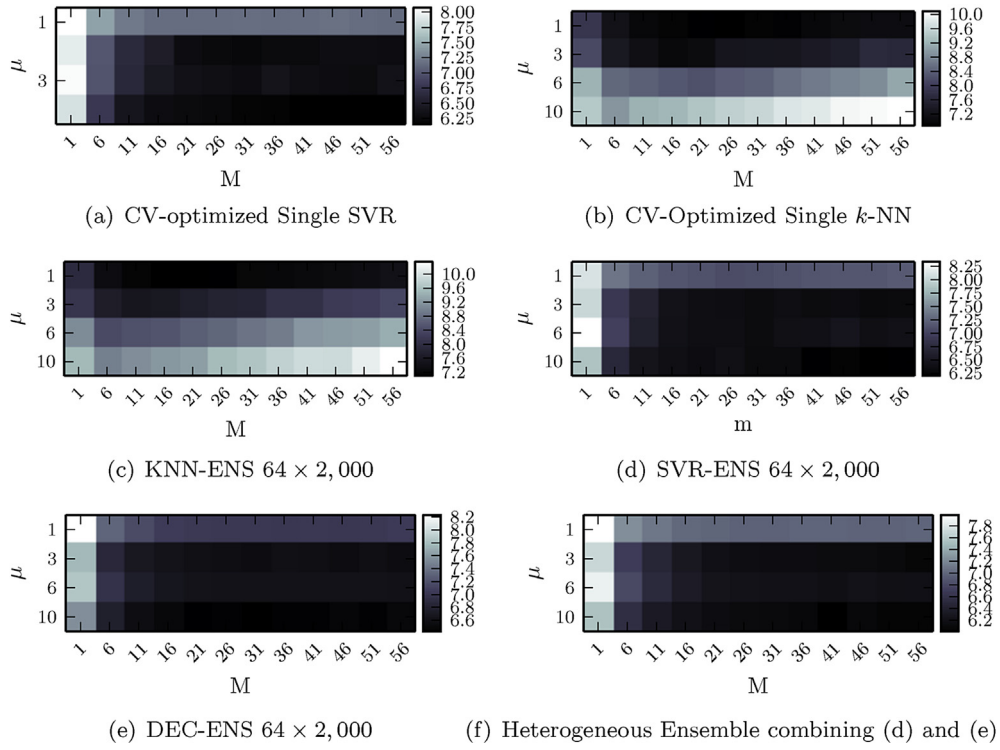


Fig. 4. Behavior of different regression algorithms for varying number of employed neighbor turbines M (x-axis) and feature window μ (y-axis) for a turbine near Vantage. The MSE is visualized by the color: A darker color denotes a lower error while a lighter color appears for higher prediction errors. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

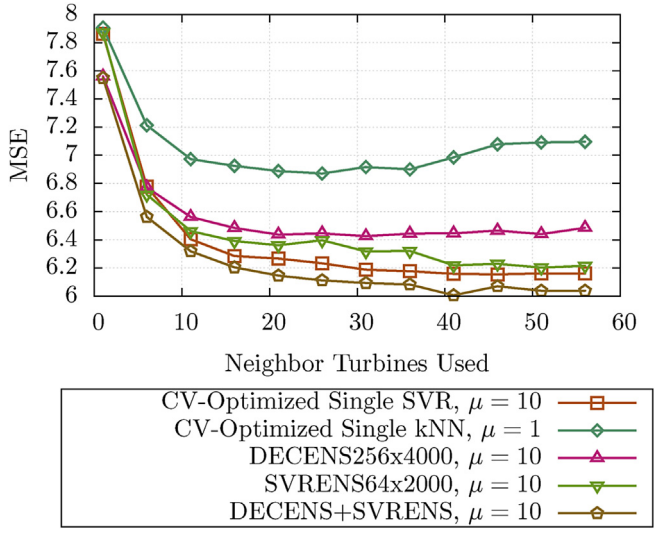


Fig. 5. Comparison of MSE depending on number M of used neighbor turbines for six regression algorithms for a turbine near Vantage.

understanding of wind and the idea behind our spatio-temporal model in mind, this team play of both parameters is a reasonable behavior. Fig. 5 shows a comparison of the six algorithms with the best setting for μ for each algorithm.

While these results show that a better prediction is possible with a larger number of considered features, we address an important issue with the selection of machine learning models in another experiment: The practitioner needs to know *a priori* which parameter values are the best for the prediction. Therefore, the parameter settings are selected on a validation set or using cross-validation technique. The question for the given task is: Can the optimal number M of neighbor turbines be selected while training the model? If this is the case, the parameter setting yielding the best training error also yields a very good if not the best prediction error on a test set. Like in Section 4, we employ a heterogeneous ensemble with 256 DT predictors using 10,000 training samples and 64 SVR predictors using 1000 training samples with a fixed feature window of $\mu = 10$.

Fig. 6 shows the MSE depending on the number $M \in \{10, \ldots, 100\}$ of neighboring turbines used for a wind turbine near Las Vegas. It can be seen that both error measurements show a similar behavior depending on M. In particular, the setting $M = 40$ yields the best error for both training and test error. The optimal found training and test errors found on five turbines are shown in Table 4. For four of the five, the value of M yielding the best training error corresponds to the same setting for the best test error. For the turbine near Casper, the best test error is achieved by a smaller number of neighbor turbines, but when using the best setting chosen by training error, the test error is still very low with 9.98. Therefore, we suggest including the number of neighbor turbines into the algorithms' parameter search. For future research we plan to conduct a more extensive study on a large number of turbines and also consider evolutionary optimization or other hyper-parameter search algorithms.

6. Power prediction for wind parks

The question comes up if the proposed prediction model can give good predictions for wind parks, too. Treiber, Heinermann, and Kramer [30] investigated the use of machine learning models for wind park power prediction while employing different aggregation settings. They found that predicting the overall power output of a whole wind park often yields a better forecast error than predicting

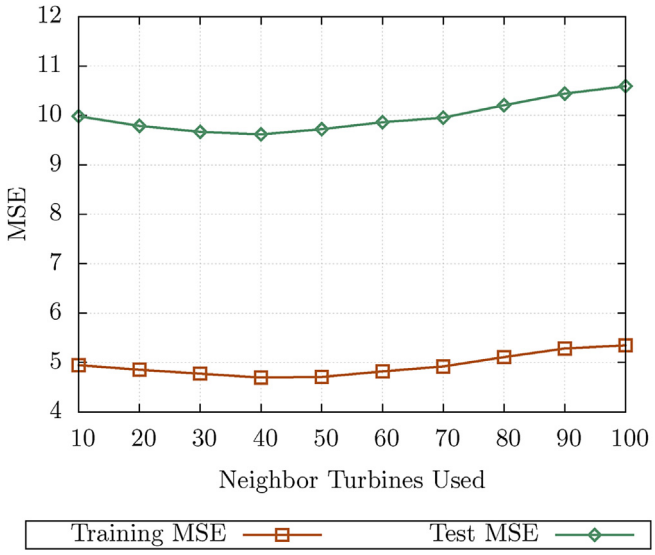


Fig. 6. Training and test error for a wind park near Las Vegas using a heterogeneous ensemble using 256 decision trees with 10,000 training samples and 64 SVR predictors using 1000 training samples. The behavior of training and test error is very similar and for both the same optimal setting (40) for the number M of neighboring turbines is found.

Table 4
Best settings for number M of used neighbor turbines selected by best training and test error.

Turbine	M	E_{train}	M	E_{test}
Casper	60	4.93	40	9.74
Hesperia	60	3.81	60	6.91
Las Vegas	40	4.69	40	9.57
Reno	20	6.16	20	13.33
Vantage	50	2.83	50	6.09

the outputs of the single turbines and summing them up. In the following, we compare our heterogeneous ensemble approach to state-of-the-art SVR used for a prediction for wind parks. Each of the five tested wind parks consists of a center turbine and 10 neighboring turbines.

Let $p_i(t)$ be the measurement of a turbine i at a time t, and $2 \leq i \leq (M+1)$ the indices of the M neighboring turbines. For a center turbine with index 1 we define a pattern-label-pair $(\mathbf{x},y)$ for a given time t_0 as

$$\begin{pmatrix} p_1(t_0-\mu) & \dots & p_1(t_0) \\ \dots & \dots & \dots \\ p_{(M+1)}(t_0-\mu) & \dots & p_{(M+1)}(t_0) \end{pmatrix} \rightarrow \sum_{i=0}^{M+1} p_i(t_0+\lambda). \quad (3)$$

Again, the parameters for SVR are sought via three-fold cross-validation. A RBF-Kernel is employed with bandwith $\sigma \in \{1e{-}5,1e{-}4,\dots,1\}$ and $C \in \{1;10;100;1{,}000;10{,}000\}$. The ensemble regressor is built up from 256 decision trees using 10,000 training samples and 64 SVR estimators using 2,000 training samples. The patterns are constructed with $\mu = 6$ and $\lambda = 3$. The results of the comparison are shown in Table 5. One can see that the heterogeneous ensemble regressor outperforms the optimized SVR model for each of the five parks tested. The lower test errors suggest that the use of a heterogeneous ensemble model can be a good choice for the overall power prediction of wind parks.

Table 5
Power predictions for the sum of five test wind parks. The test error (MSE) is given.

Center turbine	SVR with CV	Heterogeneous ensemble
Casper	888.09	870.86
Hesperia	785.09	762.76
Las Vegas	641.10	633.12
Reno	775.97	765.61
Vantage	592.34	588.60

7. Conclusions

Wind power can only be integrated into the power grid with a forecast model that yields a reliable prediction error and is efficient enough to compute the predictions in a reasonable time. After analyzing different types of ensemble predictors, we propose a heterogeneous ensemble approach utilizing both DT and SVR. In our comprehensive experimental evaluation, we show that our approach yields better results within a shorter computation time than state-of-the-art machine learning algorithms. Compared to SVR, our heterogeneous ensemble approach yields improvements of up to 37%. The runtime can even be decreased: Our approach decreases the computation time for training by factors from 1.60× to 8.78×. The trade-off between prediction performance and computation time can easily be managed by adapting the parameters like number of predictors, number of samples, and number of features used. Moreover, the number of neighbor turbines and past time steps can be increased to improve the prediction error further. In the future, we plan to implement automatic ensemble optimization and the integration of other regression algorithms into the ensemble.

Acknowledgments

We thank the ministry of science and culture of Lower Saxony for supporting us with the PhD Programme *System Integration of Renewable Energies*. Furthermore, we thank the US *National Renewable Energy Laboratory* (NREL) for providing the wind data set.

References

[1] O. Kramer, F. Gieseke, B. Satzger, Wind energy prediction and monitoring with neural computation, Neurocomputing 109 (2013) 84–93.
[2] S. Salcedo-Sanz, J. Rojo-Álvarez, M. Martínez-Ramón, G. Camps-Valls, Support vector machines in engineering: an overview, Wiley Interdiscip. Rev. Data Min. Knowl. Discov. 4 (2014) 234–267.
[3] N.A. Treiber, S. Späth, J. Heinermann, L. von Bremen, O. Kramer, Comparison of Numerical Models and Statistical Learning for Wind Speed Prediction, ESANN, 2015.
[4] L. Rokach, Ensemble-based classifiers, Artif. Intell. Rev. 33 (2010) 1–39.
[5] E. Bauer, R. Kohavi, An empirical comparison of voting classification algorithms: bagging, boosting, and variants, Mach. Learn. 36 (1999) 105–139.
[6] L. Breiman, Bagging predictors, Mach. Learn. 24 (1996) 123–140.
[7] L. Breiman, Random forests, Mach. Learn. 45 (2001) 5–32.
[8] Y. Freund, R.E. Schapire, et al., Experiments with a new boosting algorithm, in: International Conference on Machine Learning, vol. 96, 1996, pp. 148–156.
[9] D.H. Wolpert, Stacked generalization, Neural Netw. 5 (1992) 241–259.
[10] G. Brown, J. Wyatt, R. Harris, X. Yao, Diversity creation methods: a survey and categorisation, Inf. Fusion 6 (2005) 5–20.
[11] O. Kramer, N.A. Treiber, F. Gieseke, Machine learning in wind energy information systems, EnviroInfo (2013) 16–24.
[12] P. Ramasamy, S. Chandel, A.K. Yadav, Wind speed prediction in the mountainous region of India using an artificial neural network model, Renew. Energy 80 (2015) 338–347.
[13] A. Kusiak, H. Zheng, Z. Song, Short-term prediction of wind farm power: a data mining approach, Energy Convers. IEEE Trans. 24 (2009) 125–136.
[14] L. Fugon, J. Juban, G. Kariniotakis, et al., Data mining for wind power forecasting, in: Proceedings European Wind Energy Conference & Exhibition EWEC, 2008, p. 2008.
[15] J. Heinermann, O. Kramer, Precise wind power prediction with SVM ensemble regression, in: Artificial Neural Networks and Machine Learning–ICANN 2014,

Springer, 2014, pp. 797–804.
[16] S. Hassan, A. Khosravi, J. Jaafar, Examining performance of aggregation algorithms for neural network-based electricity demand forecasting, Int. J. Electr. Power Energy Syst. 64 (2015) 1098–1105.
[17] P. Chakraborty, M. Marwah, M.F. Arlitt, N. Ramakrishnan, Fine-grained photovoltaic output prediction using a Bayesian ensemble, in: AAAI Conference on Artificial Intelligence, 2012.
[18] I. Martí, M. San Isidro, D. Cabezón, Y. Loureiro, J. Villanueva, E. Cantero, I. Pérez, Wind power prediction in complex terrain: from the synoptic scale to the local scale, in: CD-Rom Proceedings of the Conference: the Science of Making Torque from Wind, Delft, The Netherlands, 2004.
[19] F. Castellani, M. Burlando, S. Taghizadeh, D. Astolfi, E. Piccioni, Wind energy forecast in complex sites with a hybrid neural network and cfd based method, Energy Proc. 45 (2014) 188–197.
[20] T. Gneiting, A.E. Raftery, A.H. Westveld, T. Goldman, Calibrated probabilistic forecasting using ensemble model output statistics and minimum CRPS estimation, Mon. Weather Rev. 133 (2005) 1098–1118.
[21] T.L. Thorarinsdottir, T. Gneiting, Probabilistic forecasts of wind speed: ensemble model output statistics by using heteroscedastic censored regression, J. R. Stat. Soc. Ser. A (Stat. Soc.) 173 (2010) 371–388.
[22] T.K. Ho, The random subspace method for constructing decision forests, Pattern Anal. Mach. Intell. IEEE Trans. 20 (1998) 832–844.
[23] T. Dietterich, Ensemble Methods in Machine Learning, in: Lecture Notes in Computer Science, vol. 1857, Springer, Berlin Heidelberg, 2000, pp. 1–15.
[24] T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning, second ed., Springer, 2009.
[25] L. Breiman, J. Friedman, C. Stone, R. Olshen, Classification and Regression Trees, in: The Wadsworth and Brooks-Cole Statistics-probability Series, Taylor & Francis, 1984.
[26] I. Steinwart, A. Christmann, Support Vector Machines, Information Science and Statistics, Springer, 2008.
[27] I. Witten, E. Frank, M. Hall, Data Mining: Practical Machine Learning Tools and Techniques: Practical Machine Learning Tools and Techniques, the Morgan Kaufmann Series in Data Management Systems, Elsevier Science, 2011.
[28] J.L. Bentley, Multidimensional binary search trees used for associative searching, Commun. ACM 18 (1975) 509–517.
[29] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: machine learning in Python, J. Mach. Learn. Res. 12 (2011) 2825–2830.
[30] N.A. Treiber, J. Heinermann, O. Kramer, Aggregation of features for wind energy prediction with support vector regression and nearest neighbors, in: European Conference on Machine Learning, Workshop Data Analytics for Renewable Energy Integration, 2013.