BeFORECAST

Deliverable: Forecast performance assessment framework

Deliverable No.: D5.1

DOCUMENT CONTROL SHEET

This project has received funding from the Energy Transition Fund of the Belgian Federal Public Service for Economy, SMEs, and Energy (FOD Economie, KMO, Middenstand en Energie)

DOCUMENT INFORMATION	
Document title	D5.1 Forecast performance assessment framework
Author(s), (organization)	Hasan Yazicioglu (3E)
Deliverable No.	D5.1
Work Package No.	WP5
Lead beneficiary	
Dissemination level	Public
Date of issue	20/11/2023

DOCUMENT HISTORY		
Version	Status	Date
V1	Final	20/11/2023

DOCUMENT APPROVAL	
Date	Author(s)
28/11/2023	Hasan Yazicioglu
Date	WP leader
28/11/2023	3E NV
Date	Coordinator
28/11/2023	VKI

DOCUMENT SUMMARY

This document summaries the developments that are done in the Task 5.1. Forecast performance assessment framework. The main outcome of this task is the release of the python package called ‘beforecast_verification’ for automated forecast verification tasks that is made available to each project partner in order to streamline the various validations.

Table of Contents

1 Introduction

Wind power, an inherently stochastic and chaotic process, poses significant challenges due to its variability and limited predictability. This unpredictability affects the planning of power grids and markets, complicates operations such as load balancing and maintenance, and introduces risks in energy trading.

Because of complex nature of wind power forecasting, development of forecast performance assessment framework becomes crucial for robust verification of results, differentiation of variations in different simulation results, interaction with previously generated results to enable refining of a particular analysis.

3E has developed and published a python package that serves certain functionalities to calculate forecast KPIs and corresponding visualizations for evaluation of forecast performance along with storage capabilities by focusing on the needs of the forecast developers.

The 'beforecast-verification' package provides improved decision-making mechanism by including various evaluation and scoring techniques to enable user to determine model representative benchmark forecast results based on operational benchmark events or ongoing model development outputs.

2 beforecast-verification

The python package 'beforecast-verification' was developed and deployed on private gitlab repository which can be installed via python-pip directly by using user specific token. The high-level architecture of the package is provided in Fig 1.

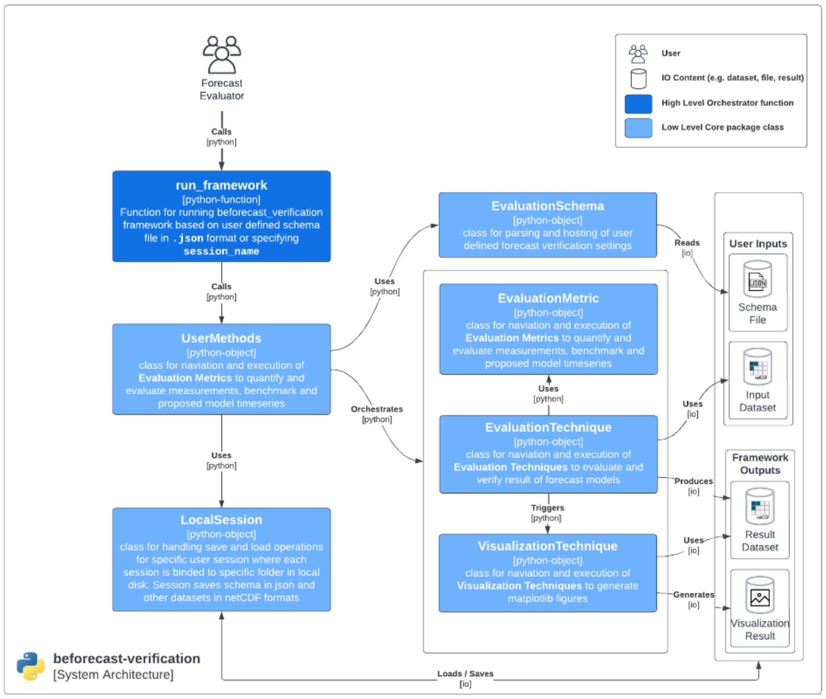

Forecast Evaluator
Calls [python]
run_framework [python-function] Function for running beforecast_verification framework based on user defined schema file in .json format or specifying session_name
Calls [python]
UserMethods [python-object] class for naviation and execution of Evaluation Metrics to quantify and evaluate measurements, benchmark and proposed model timeseries
Uses [python]
LocalSession [python-object] class for handling save and load operations for specific user session where each session is binded to specific folder in local disk. Session saves schema in json and other datasets in netCDF formats
Uses [python]
Orchestrates [python]
EvaluationSchema [python-object] class for parsing and hosting of user defined forecast verification settings
EvaluationMetric [python-object] class for naviation and execution of Evaluation Metrics to quantify and evaluate measurements, benchmark and proposed model timeseries
Uses [python]
EvaluationTechnique [python-object] class for naviation and execution of Evaluation Techniques to evaluate and verify result of forecast models
Triggers [python]
VisualizationTechnique [python-object] class for naviation and execution of Visualization Techniques to generate matplotlib figures
Reads [io]
Uses [io]
Produces [io]
Uses [io]
Generates [io]
User Inputs
Schema File
Input Dataset
Framework Outputs
Result Dataset
Visualization Result
Loads / Saves [io]
beforecast-verification [System Architecture]
User
IO Content (e.g. dataset, file, result)
High Level Orchestrator function
Low Level Core package class

Figure 1. High level architecture of the `beforecast-verification` package

The aim of the package is to provide functionalities that serve for the forecast performance evaluation. The current version of high-level framework functions is designed for execution of single forecast performance evaluation technique with specific expected input dataset format that result in evaluation technique specific result dataset. Additionally, package assets are developed to enable users for further independent orchestrations.

2.1 Evaluation metrics and techniques

Quantification of forecast evaluation provides reliability and precision for model development activities using evaluation metrics and techniques in combination. These metrics and techniques are defined in the recent publications[1][2].

Table 1. Library of evaluation metrics and techniques

Type	Forecast Evaluation	Description
Evaluation Metrics	Single Valued	Mean Error
		Mean Absolute Percentage Error
		Root Mean Square Error
		Symmetric Mean Absolute Percentage Error
		Coefficient of determination
	Binary Events	Contingency Table
		Receiver Operating Characteristic
		Brier score
		Reliability Diagram
Evaluation Techniques	Basic	Common statistics and score calculations
	Bootstrapped Aggregation	Grouping of Individual and Aggregated window

Certain categories of evaluation metrics and techniques are employed in 'beforecast-verification' and given in Table 1. Currently, evaluation metrics represent single-valued and binary events type of forecasts, and these metrics are inherited directly from xskillscore[3] library functions. Evaluation techniques are implemented by using these metrics for basic and bootstrapped aggregation techniques to be employed by the respective beFORECAST work package use cases for specific tasks. We aim to include more functionalities to support probabilistic evaluation metrics and techniques in the later phases of the project. 3E will continue to develop other categories in line with other consortium members.

2.2 Visualization library

Inventory of several visualization techniques are designed and implemented to enable the use of evaluation results as graphical devices. List of techniques is given in Table 2. Users can use specific visualization techniques independently of the high-level functions. In such cases, users should prepare the input dataset to comply with the expected format. Examples of expected formats are given in the docstring of respective visualization techniques.

Table 2. Library of provided visualization techniques

Technique Name	Purpose	Status
Bootstrapped	Convergence of candidate forecast to benchmark simulation or measurement based on temporally aggregated windows	RELEASED

Technique Name	Purpose	Status
Confidence Interval	Determination of quantile relations for certain time ranges	UNDER RELEASE
Taylor Diagram	Statistics aggregation based on standard deviation and correlation	UNDER RELEASE
Contingency Table	Feedback table of hits, misses, false alarms and correct negatives	UNDER RELEASE
ROC	Distance based forecast evaluation w.r.t. benchmark simulation or measurements	UPCOMING
Reliability Diagram	Visual representation of probabilistic contingency tables	UPCOMING

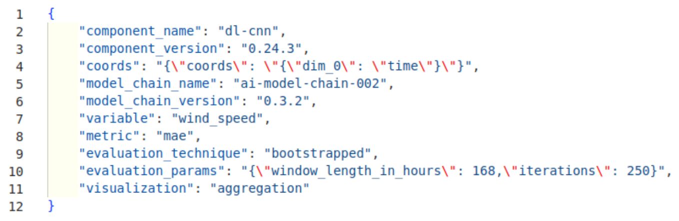
2.3 Persistent result data storage

In the framework, state preservation is achieved by implementation of a persistent data storage system to be able to perform save and load functionalities. This enables users to record specific verification results or to load any specific results back into the framework.

2.3.1 Evaluation Schema

The evaluation configurations are specified by using predefined schema which is stored as json file and provided with its dedicated json_schema file to guide the user. Example schema can be rendered using Json editor (e.g. https://json-editor.github.io/json-editor)

Certain fields are defined in the schema. We tailored the schema to its possible minimal structure by enabling user. If users desire, model related meta fields can be removed. In such case, framework will neglect these meta fields. The content of the schema is shown in Figure 2.

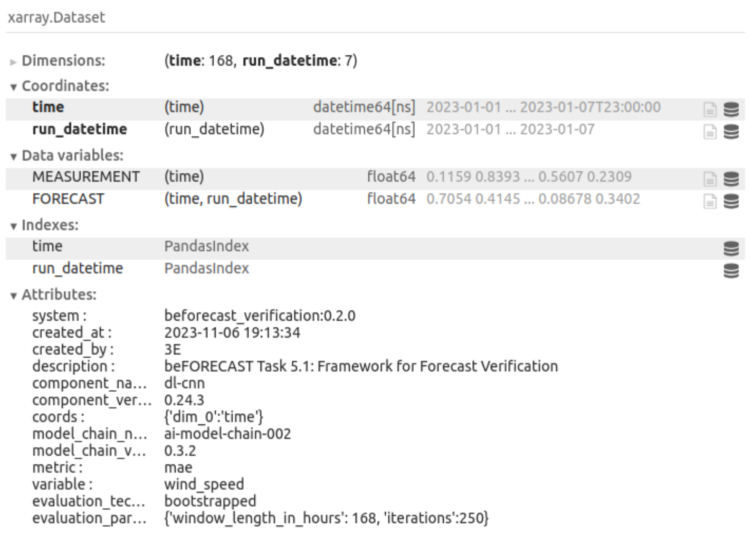


```json
{
    "component_name": "dl-cnn",
    "component_version": "0.24.3",
    "coords": "{\"coords\": \"{\"dim_0\": \"time\"}\"}",
    "model_chain_name": "ai-model-chain-002",
    "model_chain_version": "0.3.2",
    "variable": "wind_speed",
    "metric": "mae",
    "evaluation_technique": "bootstrapped",
    "evaluation_params": "{\"window_length_in_hours\": 168,\"iterations\": 250}",
    "visualization": "aggregation"
}
```

Figure 2. Example Schema by using MAE (mean absolute error) metric to evaluate wind speed predictions of model chain component w.r.t time coordinate using bootstrapped evaluation technique and producing figures by aggregation visualization technique.

2.3.2 Input data format

Input data manipulation needs to be tackled by the user. Format and use case specific examples are provided along with some utility functions. These functions aim to serialize input data into custom xarray[4] object to couple it with xskillscores[3] library and provide more meta. This allows the user to utilize the package while keeping their own internal data structure and naming conventions.



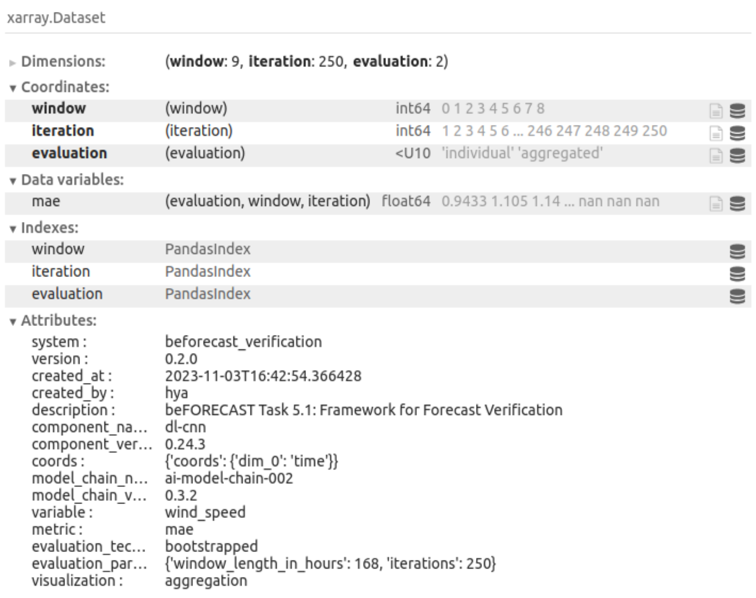
```
xarray.Dataset

▸ Dimensions:          (time: 168, run_datetime: 7)
▾ Coordinates:
    time               (time)                datetime64[ns]  2023-01-01 ... 2023-01-07T23:00:00
    run_datetime       (run_datetime)        datetime64[ns]  2023-01-01 ... 2023-01-07
▾ Data variables:
    MEASUREMENT        (time)                       float64  0.1159 0.8393 ... 0.5607 0.2309
    FORECAST           (time, run_datetime)         float64  0.7054 0.4145 ... 0.08678 0.3402
▾ Indexes:
    time               PandasIndex
    run_datetime       PandasIndex
▾ Attributes:
    system :           beforecast_verification:0.2.0
    created_at :       2023-11-06 19:13:34
    created_by :       3E
    description :      beFORECAST Task 5.1: Framework for Forecast Verification
    component_na...    dl-cnn
    component_ver...   0.24.3
    coords :           {'dim_0':'time'}
    model_chain_n...   ai-model-chain-002
    model_chain_v...   0.3.2
    metric :           mae
    variable :         wind_speed
    evaluation_tec...  bootstrapped
    evaluation_par...  {'window_length_in_hours': 168, 'iterations':250}
```

Figure 3. Example Input data format that will be used for evaluation and visualization

2.3.3 Result data format

The result data of the evaluation results are created by using xarray [4]dataset object which is fully compliant with netCDF4, the standard for climate and forecast data, requirements. Currently, the framework supports only storage over netCDF files. Sample result dataset is given in Figure 4.



```
xarray.Dataset

▸ Dimensions:          (window: 9, iteration: 250, evaluation: 2)
▾ Coordinates:
    window             (window)                          int64  0 1 2 3 4 5 6 7 8
    iteration          (iteration)                       int64  1 2 3 4 5 6 ... 246 247 248 249 250
    evaluation         (evaluation)                      <U10   'individual' 'aggregated'
▾ Data variables:
    mae                (evaluation, window, iteration) float64  0.9433 1.105 1.14 ... nan nan nan
▾ Indexes:
    window             PandasIndex
    iteration          PandasIndex
    evaluation         PandasIndex
▾ Attributes:
    system :           beforecast_verification
    version :          0.2.0
    created_at :       2023-11-03T16:42:54.366428
    created_by :       hya
    description :      beFORECAST Task 5.1: Framework for Forecast Verification
    component_na...    dl-cnn
    component_ver...   0.24.3
    coords :           {'coords': {'dim_0': 'time'}}
    model_chain_n...   ai-model-chain-002
    model_chain_v...   0.3.2
    variable :         wind_speed
    metric :           mae
    evaluation_tec...  bootstrapped
    evaluation_par...  {'window_length_in_hours': 168, 'iterations': 250}
    visualization :    aggregation
```

Figure 4. Sample result data format from bootstrapped aggregated evaluation technique

2.3.4 LocalSession object

For state preservation of the ongoing analysis, LocalSession object is introduced to enable load and save functionalities using schema, input dataset and result dataset. Implementation stores schema and runtime related fields in Json, input and result datasets in netCDF files.

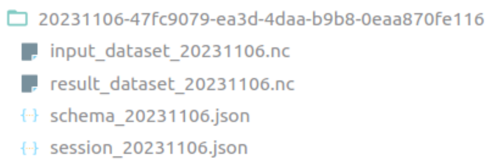

20231106-47fc9079-ea3d-4daa-b9b8-0eaa870fe116
input_dataset_20231106.nc
result_dataset_20231106.nc
schema_20231106.json
session_20231106.json

Figure 5. Sample auto generated session directory in disk location.

Session name can be provided by the user when running the framework or its name will be automatically assigned. Example of auto generated session directory is shown in Figure 5. Auto generated folder name is created by using the execution create date and random uuid.

3 Examples

To help a user get started with the package, three different use case examples are provided in a jupyter notebook (*.ipynb) format. These applications provide a step-by-step usage of the python package code including text and plots to provide more context.

3.1 Bootstrapped result aggregation

In the notebook called Example_3_1_bootstrapped_result_aggregation.ipynb, we use the output of the deep learning model of 3E which can predict wind speeds around the globe for any historical period starting from 1979 that is being developed in T2.6. A public measurement tower is selected from a tall mast inventory to represent the reference measurement dataset and 3E generated virtual timeseries on the point of measurement station. The aim in this example is to find the minimum required representative period for evaluating the 3E virtual timeseries by using the bootstrapped result aggregation as graphical device to analyze.

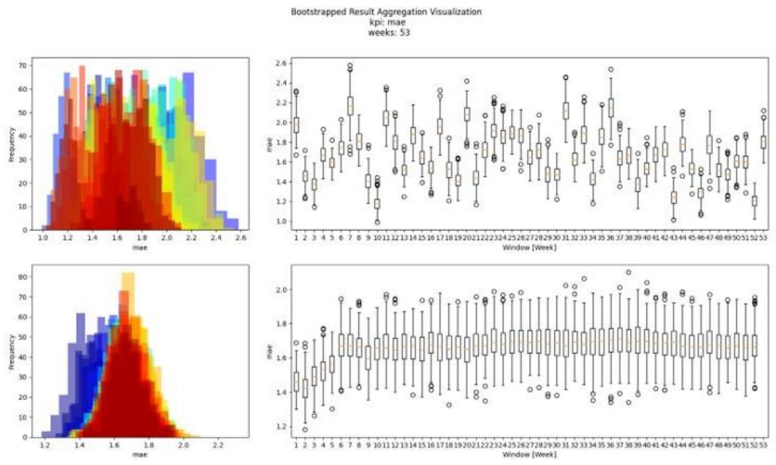


Figure 6. Bootstrapped result aggregation of forecast result w.r.t on-site measurement

Based on Figure 6, individual windows do not have much valuable information other than some notable cycling behavior. However, aggregated window representation shows that we need to use minimum length of 6-7 weeks of provided simulation output to compare it with other candidate forecasts or its variations. The reference minimum length can be defined based on end goal representative statistics. In this case, we focus on error stabilization which is calculated for mean absolute error in each weekly window.

3.2 Taylor Diagram

In the notebook called Example_3_2_taylor_diagram.ipynb, we calculated aggregated power production statistics for the entire Belgian offshore wind farms for 2 weeks of period. As comparison, we calculated expected power production from datasets of IFS_HRES, GFS and ERA5 using sample power curve.

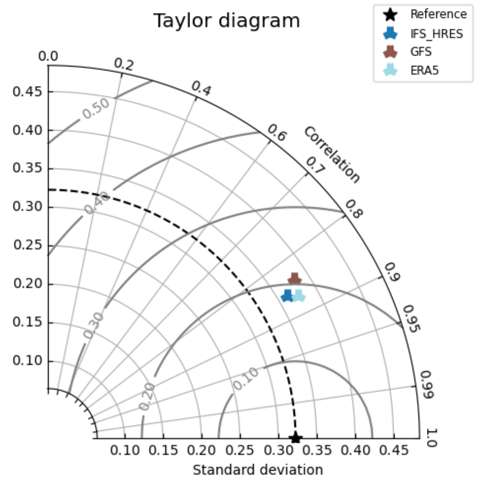


Figure 7. Taylor diagram that summarizes the statistics

Based on Figure 7, IFS_HRES is the best performing forecast compared to GFS and ERA5 datasets based on investigated time window in terms of both correlation and standard deviation.

3.3 Contingency Table

In the notebook called Example_3_3_contingency_table.ipynb, we compare various wind ramp detection events which are generated from range of ensemble members. In this case, we ensemble GFS and IFS_HRES input datasets by assigning certain weighting coefficient based on the wind direction. True ramp events are defined manually based on visual analysis and used for contingency table generation in combination of detected wind ramp events.

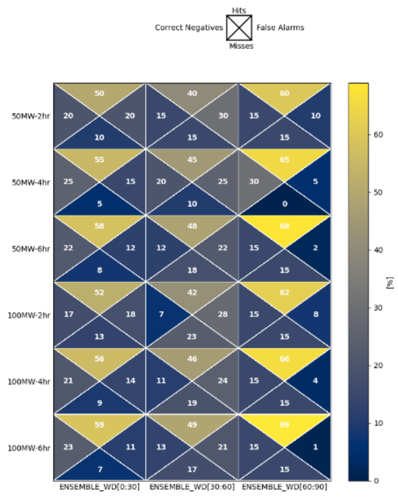


Figure 8. Contingency table to demonstrate binary event detection

Based on Figure 8, the ramp predictions of ENSEMBLE_WD[60:90] scenario are better predicted than other cases. We can detect the variations in the further model prediction by coupling framework results and evaluating the several contingency tables as graphical devices. For example, we can try predicting these events with other ensembles and we can trace the forecast performance of ramp event forecasting algorithm and quantify the skill scores of each candidate ensemble.

4 Model Evaluation

The `beforecast-verification` package will be used in the `beFORECAST` project for verification and quantification of forecast performance against benchmark forecast models and on-site measurements.

As defined in IEC36 Task[6], there are certain key considerations for forecast evaluation preparation.

A) Choice of forecast horizon: back-cast (retrospective forecast), realtime, near-realtime, intraday, day-a-head
B) Weather conditions: complex and simple weather conditions
C) Historical data/measurements: Ground truth conditions that includes seasonal variations and associated patterns
D) Representativeness: scale of aggregation single turbine, single park or multiple parks
E) Metrics: Sensitive metrics and error functions

The `beforecast-verification` is designed to be relatively simple, generic and easily extendable. Its functionalities are easily scalable for serving the respective task executions by utilizing the three base scoped stories.

4.1 Determination of benchmark simulation

The initial phase in forecast model development is to establish benchmark simulation(s). Benchmark simulation(s) can be created by composing various forecast outputs or selecting only one source based on evaluation results of various forecast outputs. Each created simulation output should be versioned and evaluated in terms of its consistency, stability and reliability to become candidate versions to be deployed or used as baseline for further development activities.

Candidate benchmark simulations ideally should be tested for diverse time ranges to account for seasonal variations in the weather conditions and patterns using historical data or meteorological measurements which are taken under complex or simple weather conditions. This assists in quantifying and addressing the modelling limitations [6][7].

To note that, designing benchmarking simulation takes time and its iterative process by nature. Its complexity can gradually be increased during the model development activities instead of focusing on its perfection.

4.2 Determination of case representative evaluation metrics and techniques

Depending on the final usage of the forecast simulation, required evaluation techniques and metrics would vary. Using benchmark simulation and historical asset data or on-site meteorological measurement case representative evaluation metrics and techniques can be determined [7][8]. In this context, `beforecast-verification` provides various evaluation metrics and techniques to quantify case representative methods as defined in Section 2.1.

4.3 Iteration of simulation model using benchmark simulation

In the context of continuous improvement and development of existing forecast models, the accuracy, precision and reliability of the forecast may deviate during the model development phase. Newer stable output needs to be versioned and verified to be able to decide acceptance or rejection of candidate versions to update the operational pipelines. Verification of the candidate versions would be done by evaluating the significance of newer versions and evolution of the forecast performance. One of the evaluation techniques that is implemented in `beforecast-verification` and proposed by IEA36[1] is to use bootstrapped result aggregation. Using this method with combination of on-site measurement and benchmark simulation, forecast evaluator can quantify the significance of the candidate version and its deviations from expected characteristics.

5 Conclusion

This document covers the functionality of the `beforecast-verification` package and illustrates its usage through examples and demonstration of use cases.

The package serves the work packages of `beFORECAST` project for task executions to enable streamlined evaluation of forecast model development.

Finally, we present model evaluation techniques that can fulfill the requirements on steps required for forecast verification.

6 References

[1] Messner, JW, Pinson, P, Browell, J, Bjerregård, MB, Schicker, I. Evaluation of wind power forecasts—An up-to-date view. *Wind Energy*. 2020; 23: 1461–1481. https://doi.org/10.1002/we.2497

[2] Joseph C.Y. Lee, Caroline Draxl, Larry K. Berg, Evaluating wind speed and power forecasts for wind energy applications using an open-source and systematic validation framework, Renewable Energy, Volume 200, 2022, Pages 457-475, ISSN 0960-1481, https://doi.org/10.1016/j.renene.2022.09.111.

[3] xskillscore: Metrics for verifying forecasts [xskillscore: v0.0.24], Github repository, Accessed at 2023.11.06 via https://github.com/xarray-contrib/xskillscore

[4] xarray: N-D labeled arrays and datasets [xarray: v2023.10.01]. Accessed at 2023.11.06 via https://github.com/pydata/xarray

[5] WE-Validate, Github repository, Accessed at 2023.11.06 via https://github.com/a2edap/WE-Validate/tree/main

[6] Möhrlen, C., Zack, J. W., & Giebel, G. (2023). Conducting a benchmark or trial. In Elsevier eBooks (pp. 89–96). https://doi.org/10.1016/b978-0-44-318681-3.00019-2

[7] Möhrlen, C., Zack, J. W., & Giebel, G. (2023a). Best practice recommendations for benchmarks/trials. In Elsevier eBooks (pp. 101–103). https://doi.org/10.1016/b978-0-44-318681-3.00021-0

[8] Möhrlen, C., Zack, J. W., & Giebel, G. (2023b). Best practice recommendations for forecast evaluation. In Elsevier eBooks (pp. 147–184). https://doi.org/10.1016/b978-0-44-318681-3.00027-1