



universität
wien

MASTERARBEIT / MASTER'S THESIS

Titel der Masterarbeit / Title of the Master's Thesis

The Added Value of Machine Learning in Forecasting Wind Turbine Icing

verfasst von / submitted by

Lukas Kugler, BSc

angestrebter akademischer Grad / in partial fulfilment of the requirements for the degree of

Master of Science (MSc)

Wien, 2019 / Vienna, 2019

Studienkennzahl lt. Studienblatt /
degree programme code as it appears on
the student record sheet:

UA 066 614

Studienrichtung lt. Studienblatt /
degree programme as it appears on
the student record sheet:

Masterstudium Meteorologie

Betreut von / Supervisor:

Ass.-Prof. Mag. Dr. Manfred Dorninger

Abstract

Icing events at wind turbines in elevated areas cause unexpected outages in power production and significant costs. More accurate forecasts would improve power production predictability and could partially reduce downtimes. Thus, advanced post-processing by machine-learning methods is applied to obtain probabilistic forecasts. The quality and improvement by using advanced methods is estimated in terms of the forecasts' discrimination, calibration and economic value. The results confirm that the proposed machine-learning methods significantly outperform empirical & logistic classification approaches for customers with relatively high cost-loss ratios.

The idea for this thesis originates from the Austrian research initiative ICE CONTROL which aims to improve icing forecasts that are commonly obtained by applying an empirical decision criterion on temperature and humidity from numerical weather forecasts. Another approach is to drive physical icing models from numerical weather prediction (NWP) models to classify whether or not there will be rotor blade icing within a certain time interval. Finally, as soon as in-situ observations exist, simple statistical tools like logistic regression can provide calibrated probabilities. Logistic regression, however, is not able to model the intrinsic multiplicative structure of icing, where multiple factors must be present at the same time to a sufficient extent to produce icing.

In this study, the added value of using machine-learning (ML) methods over simple statistical tools was quantified. Despite of ML models' higher complexity, it is not clear for them to supersede simpler models because the studied events are quite rare with occurrence rates between 5 and 25 % leading to class imbalance problems, as well as a possible lack of samples with data of only two winter seasons.

The evaluation included logistic regression, generalized additive models (GAM), support vector machines (SVM), decision trees and artificial neural networks. The NWP input data was taken from the Weather Research and Forecasting Model (WRF) which was run specifically for this project in a 2-domain configuration that downscales ECMWF's IFS model from 12.5 km to a 2.5 km resolution inner nest.

The verification was conducted using measurements that were taken directly on the nacelle of a wind turbine at a wind farm near Ellern, Germany. Three different classes of icing were used as predictand: *Meteorological icing*, defined by atmospheric temperature and dewpoint spread; *Instrumental icing*, defined by a visual classification of camera images that detects icing on the turbine; and lastly the *Visible accretion* which describes times with increasing amounts of ice on the turbine as seen by the camera.

Zusammenfassung

Die Vereisung von Windkraftanlagen verursacht unerwartete Ausfälle der Stromproduktion und nennenswerte Folgekosten. Da genaue Vorhersagen die Planbarkeit der Stromerzeugung erhöhen und die Stillstandszeiten reduzieren würden, werden in dieser Arbeit 'Machine-Learning' Methoden angewandt. Die Qualität und Verbesserung der Vorhersagen durch die Nutzung dieser Methoden wird auf drei Arten bestimmt: Die Fähigkeit Vereisung von Nicht-Vereisung zu unterscheiden (Diskriminierung), die Fähigkeit kalibrierte Wahrscheinlichkeiten vorherzusagen (Kalibrierung), sowie am relativen ökonomischem Wert (PEV). Die Ergebnisse zeigen signifikante Steigerung an ökonomischem Wert durch den Einsatz der betrachteten Methoden für Nutzer mit relativ hohem C/L-Verhältnis im Vergleich zu empirischer & logistischer Klassifikation.

Diese Arbeit wurde durch das österreichische Forschungsprojekt 'ICE CONTROL' angestoßen, welches zum Ziel hat, Vereisungsprognosen zu verbessern. Diese werden üblicherweise aus den Prognosen der numerischen Wettervorhersage (NWP) generiert, indem ein Entscheidungskriterium auf Temperatur und Luftfeuchte angewandt wird. Alternativ werden auch physikalische Vereisungsmodelle mit den Randbedingungen aus der NWP angetrieben, um zu berechnen, ob es in einer bestimmten Zeitspanne zu Rotorvereisung kommt oder nicht. Zu guter Letzt können, sobald Vor-Ort-Messungen vorliegen, auch statistische Modelle wie logistische Regression angewandt werden, um kalibrierte Wahrscheinlichkeiten vorherzusagen. Logistische Regression ist jedoch nicht in der Lage, die intrinsisch multiplikative mathematische Struktur der Vereisung zu modellieren, in der mehrere Faktoren gleichzeitig in ausreichendem Maße vorhanden sein müssen, um Vereisung zu bewirken.

In dieser Arbeit wurde der Mehrwert durch die Nutzung von Machine-Learning (ML) Methoden gegenüber simplen statistischen Modellen quantifiziert. Die höhere Komplexität der ML Modelle ermöglicht zwar grundsätzlich bessere Vorhersagen im Vergleich zu einfachen Modellen, es ist jedoch nicht offensichtlich, dass dies auch in diesem Fall so ist. Die Seltenheit der Vereisung (5-25 %) führt zu Problemen aus dem Missverhältnis der Klassen (Vereisung/Nicht-Vereisung) und einem Bedarf an längeren Zeitreihen. Die aktuelle Zeitreihe mit Daten aus zwei Wintern könnte ein Hindernis darstellen.

Die Ergebnis-Evaluierung beinhaltet logistische Regression, generalisierte additive Modelle (GAM), Support-Vector-Machines (SVM), Entscheidungsbaum-basierte Modelle, sowie künstliche neuronale Netze. Die NWP Daten stammen aus dem Weather Research and Forecasting Model (WRF), welches eigens für dieses Projekt zum 'Downscaling' des IFS Modells (ECMWF) auf 2.5 km Auflösung herangezogen wurde.

Die Verifikation wurde mit Messungen auf der Windrad-Nabe durchgeführt. Drei unter-

schiedliche Klassen der Vereisung wurden definiert: Meteorologische Vereisung, mittels Temperatur und Taupunktsdifferenz; Instrumentelle Vereisung, mittels visueller Klassifikation vor Ort; sichtbare Akkretion, durch eine sichtbare Zunahme an Eismasse.

Acronyms & Abbreviations

AUC	see AUROC
AUROC	Area under the ROC curve
BSS	Brier skill score
Bagging	Bootstrap aggregation
Boosting	A meta-algorithm for decision trees which sequentially fits trees on the residual of the previous tree's prediction
-	
Bootstrap aggregation	A meta-algorithm for decision trees which independently fits trees on bootstrapped samples
-	
CART	Classification and regression trees
CDF	Cumulative distribution function
CV	Cross-validation
DMO	Direct model output
Deep learning	The subdomain of applying neural networks with more than one hidden layer
ECMWF	European Centre for Medium-Range Weather Forecasts
Feature space	The space spanned by the predictor variables
GAM	Generalized additive model
Hyperparameters	Model parameters which need to be set before training the model; they can be estimated by cross-validation
-	
IFS	Integrated Forecasting System
LHS	left hand side (of the equation)
Learning	Abbreviation for machine learning
ML	Machine-Learning
MSL	(above) mean sea level
PEV	Potential economic value
RHS	right hand side (of the equation)
SVC	Support-Vector-Classification
SVM	Support-Vector-Machine (umbrella term for SVC & SVR)
SVR	Support-Vector-Regression
Training	The process of learning the model's transfer function
VC	Vapnik-Chervonenkis
WRF	Weather Research and Forecasting Model

Contents

1	Introduction	1
1.1	Problem statement	2
1.2	Contributions	2
1.3	Mechanisms of icing on structures	3
1.4	Related work	3
2	Data description	6
2.1	Observation data	6
2.2	Forecast data	8
3	Methods	10
3.1	Statistical learning theory	10
3.1.1	Bias-variance tradeoff	13
3.1.2	Estimating out-of-sample performance in practice	17
3.2	Prediction methods	19
3.2.1	Logistic regression	20
3.2.2	Generalized additive models	21
3.2.3	Artificial neural networks	22
3.2.4	Classification and regression trees	24
3.2.5	Support vector machines	25
3.3	Verification	29
3.3.1	ROC & Reliability diagram	29
3.3.2	Brier score	30
3.3.3	Economic value score	31
4	Results	36
4.1	Empirical icing classification of direct model output	37
4.2	Physical icing models and logistic regression	39
4.3	Adding nonlinearity: Generalized additive models	42
4.4	Artificial neural nets: An entirely nonlinear approach	48
4.4.1	Resampling and regularization to address class imbalance problems	54
4.5	Classification trees and support vector machines	55
5	Discussion and Outlook	59
Bibliography		63
List of Figures		65
List of Tables		66
A Appendix		68

1 Introduction

Icing of wind turbines poses a considerable health and safety risk from ice throw and is responsible for economic damage. When icing occurs, rotor blades accumulate sheets of ice, which break off the surface at some stage and get thrown hundreds of meters (Davis et al., 2014b) from the turbine site due to the rotor's height and momentum. The uneven structural load from ice is furthermore suspected to reduce the turbine's lifetime (Frohboese and Anders, 2007). These threats are easily minimized by stopping the rotor from spinning, reducing wear and minimizing the distance of the thrown ice. On the other hand, the reduced power production requires energy companies to compensate for this loss with other powerplants which can be costly.

Even for locations where ice throw is not an issue, continuing operation under icing conditions is not preferred since the detrimental effect on the aerodynamics of the rotor blades leads to a significant drop in power production compared to no-icing conditions.

The present work aims to provide improved forecasts of wind turbine icing by postprocessing the output of numerical weather prediction models with observed data using statistical and machine-learning models, comprising of algorithms like generalized additive models, artificial neural nets, support vector classification and decision trees.

This work was made possible through the participation of the Department of Meteorology and Geophysics (IMGW) in the ICE-CONTROL project (2016-19), which provided data and other resources.

Motivation Operational turbines suffer from shortening of component lifetimes, decreased annual energy production (Davis et al., 2014b), changed aerodynamic forces (Jasinski et al., 1998), which, together with additional mass on the blades can lead to increased loads on the turbine (Ronsten et al., 2012) and increased maintenance costs. Shedding ice from blades can lead to structural imbalance (Frohboese and Anders, 2007) and increased safety risks due to ice throw. Increased aerodynamic drag leads to reduced power production (Virk et al., 2010).

More accurate predictions on the duration and ending of icing-conditions allow wind park operators to restart power production earlier by deicing the rotor blades with built-in heating elements. Power companies also reduce their financial risks by increased predictability of power production in day-ahead pricing while it is estimated that 24% of the globally installed wind energy capacity is located in a climate that is vulnerable to icing for a large part of the year (Davis et al., 2014b).

1.1 Problem statement

While machine-learning methods have been used in countless domains, there is no study on whether these methods can improve the prediction of this specific task due to several difficulties that arise:

- Wind-turbine icing can be quite a rare event depending on the location of the site and the definition of icing, ice accretion for example was only observed in 5% of all samples. Rare events have less impact on the cost function of a classifier, such that the resulting decision is biased towards the majority class (non-icing). This becomes apparent in reduced detection and false-alarm rates (see section 4.4.1) if the imbalance is not accounted for.
- Time-series of in-situ measurements on wind-turbines tend to be critically short. Even two years of data capture only a low number of icing periods, i.e. positive events. To obtain data of more than one hundred icing periods, it could be necessary to take measurements for several years, which involves significant costs for installing and continuously operating the in-situ measurements. Statistical learning theory ([Abu-Mostafa et al., 2012](#)) tells us that for a given number of samples, an increase in model-complexity will result in worse predictions (see section 3.1). Therefore it is unclear beforehand, whether a more complex model improves the predictions.

1.2 Contributions

In order to improve the prediction of wind-turbine icing by means of machine-learning, this thesis focuses on estimating

- the impact of using more and more powerful post-processing tools compared to simpler approaches,
- the relative and absolute performance of each ML method in terms of discrimination, economic value and reliability together with
- the uncertainty of each metric.

More specifically, this thesis answers the following questions:

- For each model: Can we improve predictions by using a more complex¹ model? How much gain is there and under which conditions, i.e. high/low sensitivity to false alarms?

¹A complex model in terms of statistical learning theory typically has many degrees of freedom but it does not have to be hard to understand. See section 3.1 for the definition of complexity.

- What are strengths and weaknesses of the methods?
- How stable are the models in their predictive performance under variations in the input data that simulate the variability of weather regime? The dataset of two winters is resampled to simulate the natural variability of weather conditions.

Finally, difficulties and failures of machine-learning shall be documented.

1.3 Mechanisms of icing on structures

Before diving into modeling, it is useful to notice that there are several ways how ice can build up on structures. Three types of freezing include (adapted after [Andersen et al. \(2011\)](#)):

- **Accretion by condensation of water vapor and subsequent freezing**
This type of icing does not account for a significant part of wind turbine icing for the Ellern windpark in central Germany ([Weissinger, 2017](#)). Wind turbines in colder climates are expected to experience even less accretion by condensation due to lower saturation water content at lower temperatures.
- **In-cloud icing** is characterized by droplets of less than 0.1 mm colliding with the surface of the rotor blades and subsequent freezing. Most droplets in sub-zero air temperature freeze instantly upon contact, since they are already in a supercooled state.
- **Precipitation icing** requires sub-zero surface temperatures.

The first attempts to operationally predict wind turbine icing used the Makkonen model ([Makkonen, 2000](#)), which focuses on in-cloud icing as most dominant process. The same is also the case for the ICE CONTROL dataset ([Weissinger, 2017](#)).

1.4 Related work

The first contribution to the direct prediction of ice on wind-turbines was made by [Makkonen \(1984, 2000\)](#) who derived a physically based, semi-empirical model to explain the accretion of ice on power lines, since this is an important topic in the polar climates. [Drage and Hauge \(2008\)](#) used this model to simulate icing based on boundary conditions from NWP. [Thompson et al. \(2009\)](#) introduced a new physics module into the WRF model which outputs the quantities that are necessary to predict ground-structure or aircraft icing with Makkonen's model. To allow a more realistic simulation of ice load on a wind turbine, [Davis et al. \(2014a\)](#) extended the model to account for increased wind velocity due to the rotating blades, for ablation to remove ice and

for sublimation processes. [Weiβ \(2018\)](#) provides a sensitivity study of different model parameters. [Davis et al. \(2014b, 2016\)](#) also started using decision-tree models and generalized additive models (GAM) to predict relative power production using NWP output. This can be seen as the first machine-learning (ML) application in icing-prediction. Lately, [Scher and Molinder \(2019\)](#) evaluated the prediction of icing-related power production loss by using random forests, a specific ensemble of decision-trees that use the 'Bagging' approach (see 3.2.4).

In the environmental sciences, [Hsieh \(2009\)](#) noted a rise in usage of ML methods in the 1990s but it was not until the 2010s before ML finally started to have impact on a larger number of fields through the explosion of data generated and stored. Another reason could be that in the meantime, several techniques (see regularization 3.1.1) and programming interfaces were invented that eliminated pitfalls that arise in training the models and overall eased the use of these models so that a wider range of users could apply them. Since then ML methods have been used in satellite data processing, general circulation models ([Hsieh, 2009](#)), NWP postprocessing ([Rasp and Lerch, 2018](#)) and tested for climate prediction ([Rasp et al., 2018](#)).

Class imbalance From a machine-learning perspective, the present work features an additional challenge that is most easily introduced by the following example: Consider the task of (binary) predicting a rare event with climatological probability of 0.1 %. Any model can be right in 99.9 % without having learned anything by just predicting the majority class.

Since in many cases it is more important to detect the rare minority class, the goal is to increase detection rates while accepting to loose some correct rejections to false alarms. Up to now, two ways have been established to deal with this problem:

- Synthetic resampling, which picks samples of the minority class more often or ones from the majority class less often; or
- class weighting, which gives more weight to minority class samples so that they have a fair share in the cost function that is minimized in the learning process.

Presently it does not seem that there is one perfect technique to solve the misrepresentation problems that come with class imbalance ([Haixiang et al., 2017](#)). In fact, many authors have shown that class imbalance itself is not the responsible for bad classification performance:

There is even an entire research field among computer scientists that describes the problems that are often encountered with class imbalance. Starting in the late 1980s, [Holte et al. \(1989\)](#) identified 'small disjuncts' as problematic features of how a learning model classifies subspaces of the feature space. They approached the issue from a perspective similar to decision trees that is called 'concept learning'. A nice summary of the stumbling blocks involved was given by [Provost \(2000\)](#) who also raised important questions. An introduction of the 'small disjunct'

problem can be found in G. Weiss' section in the textbook of He and Ma (2013). Finally, López et al. (2013) provide a review of the issues and point out that research should find new measures of data characteristics in order to find appropriate solutions. To date, no practical techniques have been found, so that one has to keep using synthetic resampling and/or class weighting that were listed above.

Section overview

In section 2, there is an overview of the forecast and observation data used to train and validate the models and some specialities/technicalities of turbine icing is described together with already existing physical deterministic models. Section 3 introduces some statistical learning theory, all models and verification methods used in the results section. The results section 4 first establishes baseline prediction performance scores to which later models are compared and then increases the complexity of the model, finally reaching artificial neural nets and decision trees which are models of high complexity. The discussion, section 5, reviews the results and offers an outlook to what could be investigated in the future.

2 Data description

A post-processing study critically depends on both predictor and predictand data. The predictor data consisted of variables from NWP models as well as variables from the Makkonen physical icing model. As predictand variables served various observed quantities.

2.1 Observation data

All observational data was taken at one wind turbine located at 49°58'42"N, 7°40'56"E, 645 m MSL near Ellern on the Hunsrück, a low mountain range in Germany, up to 350 m above the surrounding flat topography. The number of measured quantities on the nacelle at 780 m MSL is quite large and only a few of them were retained for this study, as only those variables that form a predictand variable, i.e. an icing class, are of interest.

Phases of icing

The present study aims at predicting three phases of icing following the loose classification of [Lehtomäki \(2016\)](#), visualized in figure 1. They are based on temperature and dewpoint measurements as well as images from cameras mounted on top of the nacelle.

The exact definition of the icing classes is as follows:

- **Meteorological icing** is present when the ambient conditions allow ice accretion. For model evaluation, one needs to define concrete thresholds for the relevant meteorological variables (i.e. temperature, liquid water content, relative humidity, etc.) to classify the period as meteorological icing period. Within this study, situations with temperature $< 0^{\circ}\text{C}$ and dewpoint spread $< 2 \text{ K}$ are defined to be meteorological icing conditions.
- **Instrumental icing** is present if signs of icing on the nacelle are visible on the camera images.
- **Visible accretion** is present if the camera indicates increased ice load on the nacelle since the last observation. After [Lehtomäki \(2016\)](#), this phase is a subset of the instrumental icing phase.

The observational detection of instrumental and visible icing depends critically on the human observer who has to classify the images regardless of lighting (day/night) or visibility (fog/in-cloud) conditions. Nevertheless, since direct ice load measurements are subject to systematic errors, it is assumed that camera detection provides the best results ([Schneider, 2017](#); [Weiβ, 2018](#)).

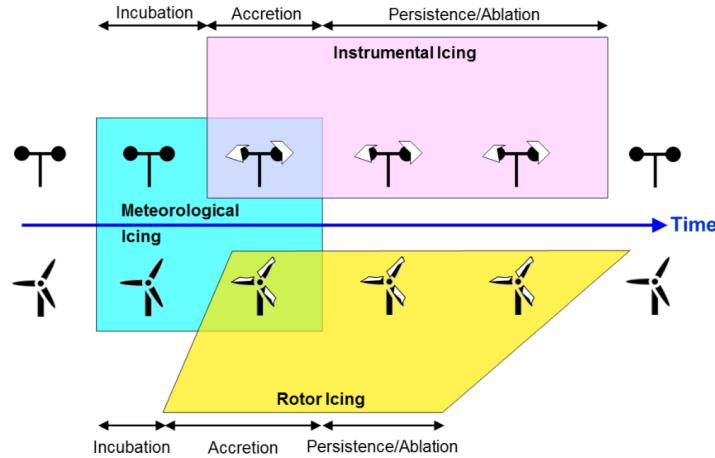


Figure 1: Definition of icing phases after [Lehtomäki \(2016\)](#)

Measurement methods For the present work, only data observed on the nacelle was used: For the meteorological icing class, temperature and dewpoint spread as measured by the Rotronic *HygroClip2 Advanced* probe and for instrumental and visible accretion phases, the icing classification timeseries provided by Meteotest. The latter phases were observed in multiple categories from which all severity classes other than no-icing were regarded as icing.

The observed variables and their respective measurement instruments are listed in table 1. The camera image analysis classification was supplied by Meteotest GmbH to the ICE-CONTROL project. The classification originates from visible images of three cameras that were installed on the turbine hub to better assess icing on the instruments on the nacelle and on the blades. Meteotest categorized the images based on the amount of visible ice into six categories and depending on the temporal change into four categories.

The data was resampled at an interval of 3600 s with an averaging interval of 600 s where the average is taken from the last time stamp to the current one, following the convention of SYNOP data. The dataset including many other observed variables was created by Lukas Strauss.

Measured quantities	Instrument	Data type and range
Temperature, dewpoint spread	Rotronic <i>HygroClip2 Advanced</i> probe	continuous, /
Instrumental icing	Meteotest camera image analysis	discrete, 0-5
Visible accretion ²	Meteotest camera image analysis	discrete, 0-3

Table 1: Observational data: variables and their respective measurement instruments.

²The original name within the dataset was 'meteorological icing', which was changed to 'visible accretion' to follow the definitions of [Lehtomäki \(2016\)](#). Meteorological icing within this work means temperature < 0 °C and spread < 2 K.

2.2 Forecast data

The NWP data originated from a multi-physics ensemble by the use of the Weather Research and Forecasting model (WRF) version 3.9 driven by the global IFS model of the ECMWF. The ensemble used 10 different parameter settings for boundary-layer, surface-layer, microphysics, land-surface-model parametrizations and land-use data. The 72 hour lead-time control run which was extensively used in this work, used Mellor–Yamada–Nakanishi–Niino boundary and surface layer parametrization, Thompson & Eidhammer aerosol aware microphysics, Rapid Update Cycle (RUC) land-surface model and MODIS land-use data. It furthermore closely follows the configuration described in [Benjamin et al. \(2016\)](#). All ensemble members feature a two-domain configuration (see figure 2), dynamically downscaling the global model data from 12.5 km to 2.5 km resolution in the inner nest with 51 vertical levels and cold-start initializations at 00 UTC every day. The outer domain spans a north-south distance from the Shetland Islands to the island of Corsica and in west-east direction from around 100km west of Ireland to western Ukraine. The inner domain spans from west Belgium to east Germany and from the northern Netherlands to the southernmost parts of Germany.

The Makkonen icing model that ingested the meteorological boundary conditions of WRF calculated the ice load (kg/m) for a standardized cylinder that should resemble a rotor blade.

Both models were run specifically for the ICE-CONTROL project. The subset of forecast variables that were used in this thesis is listed in table 2.

A verification of DMO with hub-height measurements by Lukas Strauss showed no excessive biases in temperature or dewpoint as the inner domain of the control run had less than 1 K bias in most forecast hours, yet, a diurnal cycle of about 0.5 K can be seen in temperature and a almost linear increase in dewpoint bias from about -0.8 K to +0.2 after 66 forecast hours.

Variable	Unit
Temperature	°C
Dewpoint spread	°C
Wind speed	m/s
Makkonen ice-load	kg/m
Makkonen ice-rate	kg/m/h
Cloud base height	m MSL

Table 2: The used input variables including derived ones and their physical units

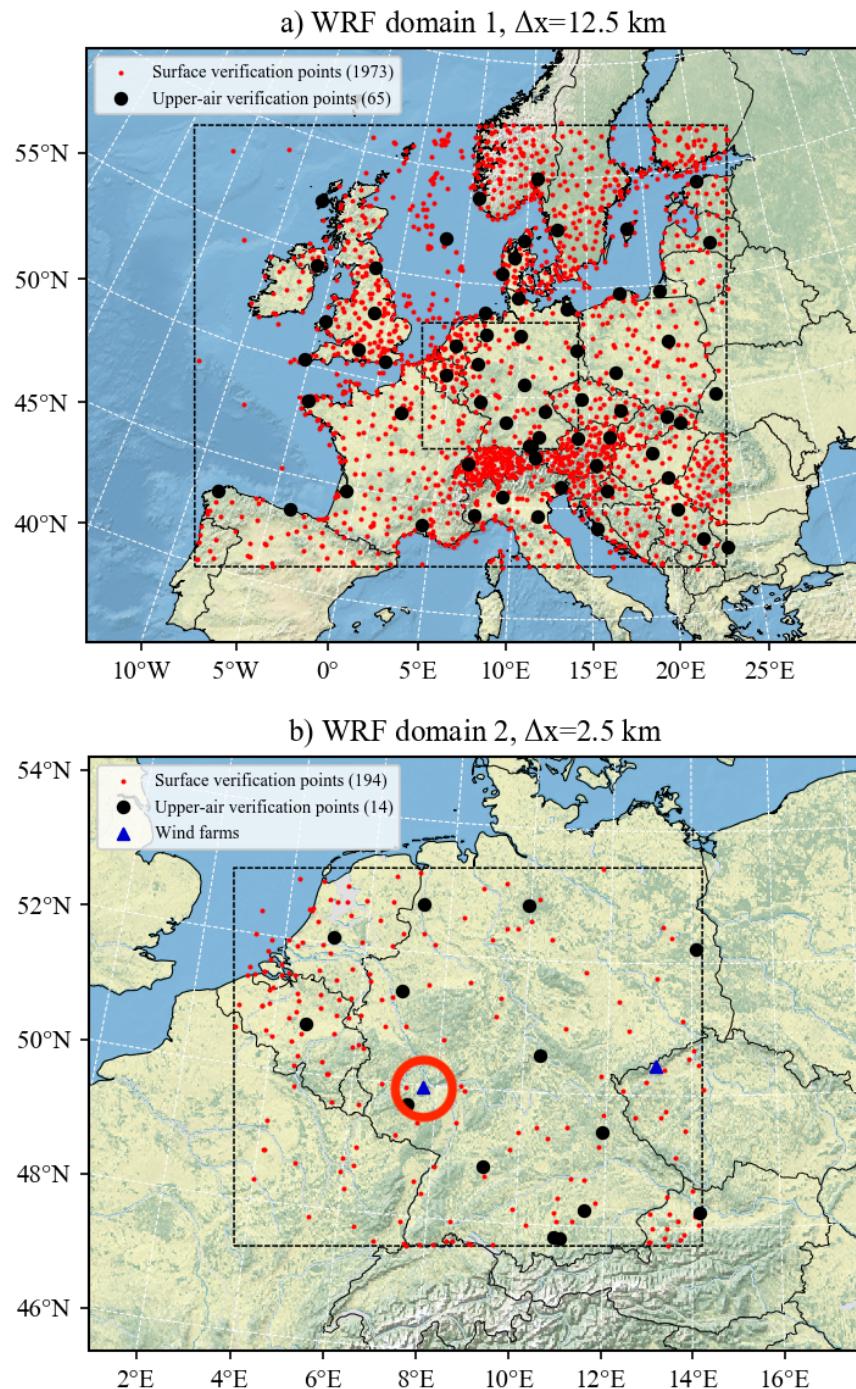


Figure 2: The outer WRF model domain and inner WRF model domain had a grid resolution of 12.5 km and 2.5 km respectively. The location of the wind turbine is marked by a red circle. Reproduced from [Strauss et al. \(2019\)](#), courtesy of Stefano Serafin.

3 Methods

Machine-learning, more specifically *supervised learning*, is the process of learning a pattern (i.e. a mathematical mapping) from samples of input and output. It is a common problem in many fields. Figure 3 gives an overview of the task at hand: On one side, there are forecast timeseries of temperature, dewpoint, windspeed and cloud base height from the NWP model. On the other side, there is a timeseries of binary values that describe whether or not there was icing in each hour. The challenge is now to find a postprocessing that maps the DMO to a binary forecast that approximates the binary observation as close as possible. The comparison of forecasts and observations is realized in the verification step. A postprocessing model learns from its forecast errors by minimizing a cost function that is different for each model.

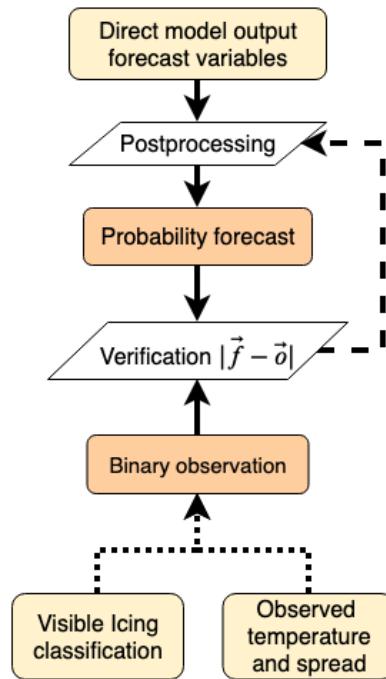


Figure 3: Overview of machine-learning: To improve postprocessing, forecast errors need to be translated into changes of the postprocessing model.

3.1 Statistical learning theory

The theoretical grounding for machine-learning (ML), apart from classical statistics, is rather young. Most of the theory in common textbooks nowadays stems from theoretical works starting

in the 1960s³ - a time where researchers used computers to solve data problems for the first time. In these years, Vapnik & Chervonenkis developed their *Statistical Learning Theory*⁴ in which they formulated the fundamental problem in precise terms. They derived a *risk functional* following their *structural risk minimization principle* from which all major supervised learning problems can be deduced as special cases of this fundamental formula: classification, regression and density estimation (Hsieh, 2009). Yet it took until 1974 for the theory to be translated from Russian into English.

In the meantime in 1984, the british computer scientist L. Valiant proposed an alternative approach of mathematical analysis to the same topic: The *Probably approximately correct learning* (PAC learning). The name of this theory may seem dubious but it describes equation 3.1 that follows a few paragraphs below quite well.

Main results The most important result from Statistical Learning Theory (SLT) for practical application is that

- the model complexity should not be too high, and
- the number of samples should not be too low.

This statement seems logical and self-explanatory but it is neither precise nor explained. Before this can be done, a few definitions will be made.

Problem formulation Vapnik (1999) describes the problem of learning from data with three components that describe how predictor, predictand and the learning model are related from a mathematical point of view. The three components are (Vapnik, 1999):

1. A generator of vectors x , drawn independently from a fixed but unknown distribution $P(x)$;
2. A supervisor that returns an output vector y for every input vector x , according to a conditional distribution function $P(y|x)$, also fixed but unknown;
3. A learning machine capable of implementing a set of functions $f(x, \alpha)$, where α are pre-defined ("hyper")parameters.

³from Vovk et al. (2015); original publication: Vapnik, V.N., Chervonenkis, A.Y.: Об одном классе алгоритмов обучения распознаванию образов (On a class of algorithms for pattern recognition learning, in Russian, English summary). Автоматика и телемеханика (Automation and Remote Control) 25(6), 937–945 (1964)

⁴a summary can be found in Vapnik and Chervonenkis (2015)

Vapnik (1999) furthermore states: "The problem of learning is that of choosing from the given set of function, the one which predicts the supervisor's response in the *best* possible way. The selection is based on a training set of l random independent identically distributed (i.i.d.) observations, $(x_1, y_1), \dots, (x_l, y_l)$, drawn from $P(x, y) = P(x)P(y|x)$."

Generalization Now it is time to introduce the main challenge that is associated with the problem of learning: Which is *the best function*? What can be observed in many cases is that a model is very accurate on historical data, but completely fails in predicting the future. This observation is nicely expressed in a phrase that is quoted to Niels Bohr: "Prediction is very difficult, especially about the future." and quantitatively put on paper in one main relation: the Hoeffding inequality, one form of the 'law of large numbers' (Abu-Mostafa et al., 2012).

The equations are formulated using four concepts:

- the **in-sample error** E_{in} , describing the model's error in approximating the outcomes in the training data that was used to build the model;
- the **out-of-sample error** E_{out} , describing the prediction error on unseen data of the same distribution;
- the **model complexity**, which is described by the *growth function*⁵ M ,
- the **generalization error** means the difference between E_{in} and E_{out} . A large generalization error is equivalent to *overfitting* the training set,

and the number of samples N .

Model complexity As described above, the task of learning is to choose a function from a set of functions (also called hypotheses). A simple set of functions is the set of all *linear* functions. These build the hypothesis set for the linear regression model. A more complex model for example is the set of all polynomials of order ≤ 2 , which also includes the order 1. Consequently, one could simply say that the larger the hypothesis set is, the larger is also the model complexity. Nevertheless, the larger the number of hypothesis, the larger is the probability of picking the wrong hypothesis - the generalization error. For a precise definition of complexity, Abu-Mostafa et al. (2012) is a good starting point.

⁵Another complexity measure for example is the *VC-dimension*.

Theoretical measures of generalization The Hoeffding inequality, one form of the 'law of large numbers', describes the risk of overfitting on the training set, i.e. having a small in-sample error and a large out-of-sample error (Abu-Mostafa et al., 2012):

$$P[|E_{in}(h) - E_{out}(h)| > \epsilon] \leq 2M \exp(-2N\epsilon^2) \quad \text{for any } \epsilon > 0 \quad (3.1)$$

The difference of errors on the left side of the equation stands for the generalization error. The point of the LHS is to describe the probability of a bad event to happen, namely that the generalization error is larger than the error tolerance ϵ . The RHS bounds this probability by a measure of model complexity (M) and the number of samples (N). Both errors depend on the hypothesis h examined. This formula is a good example of the *Probably approximately correct* (PAC) theory referenced a few paragraphs above: The statement is one of probability, thus "Probably"; and "approximately correct" because the out-of-sample error is not exactly equal to the in-sample error, but follows it closely (by ϵ) with probability $1 - P$ (P is the probability of the bad event to happen).

What one can learn from this formula, is that the probability of overfitting is reduced in an exponential way for more and more samples and that increased model complexity increases the risk.

3.1.1 Bias-variance tradeoff

Another perspective on the generalization error gives the Bias-Variance decomposition. It decomposes the squared error of a model prediction to analyze the relation between underfitting (fig. 4a) and overfitting (fig. 4b).

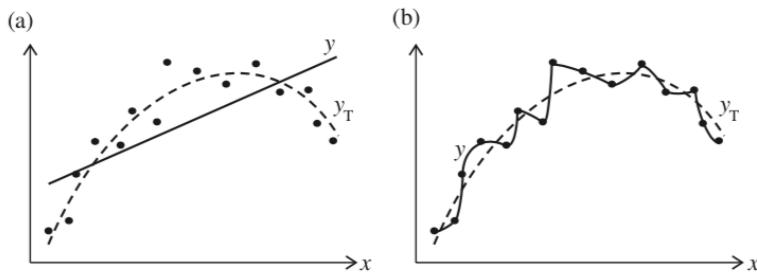


Figure 4: The bias-variance tradeoff formalizes the balance between under- and overfitting. These two examples show the solid regression function $y(x)$ trying to predict the dashed function $y_T(x)$. The model is of too low complexity in a) and too high complexity in b). Taken from Hsieh (2009).

The decomposition involves expected values over 'many' datasets from the same distribution (i.e. $\mathbb{E}_{\mathcal{D}}$) and over the distribution of input vectors \mathbf{x} (i.e. $\mathbb{E}_{\mathbf{x}}$) based on the probability distribution on the input space \mathcal{X} (Abu-Mostafa et al., 2012). The true function is $f(\mathbf{x})$ and the model prediction is $g^{(\mathcal{D})}(\mathbf{x})$. The prediction $g^{(\mathcal{D})}(\mathbf{x})$ depends on the values of the predictor variables \mathbf{x} and implicitly on the training dataset which is indicated by the superscript \mathcal{D} . The training dataset usually determines empirical parameters or the functional form of the prediction function.

The squared error can be decomposed as follows:

$$\begin{aligned}\mathbb{E}_{\mathcal{D}} \left[E_{\text{out}}(g^{(\mathcal{D})}) \right] &= \mathbb{E}_{\mathcal{D}} \left[\mathbb{E}_{\mathbf{x}}[(g^{(\mathcal{D})}(\mathbf{x}) - f(\mathbf{x}))^2] \right] \\ &= \mathbb{E}_{\mathbf{x}} \left[\mathbb{E}_{\mathcal{D}}[(g^{(\mathcal{D})}(\mathbf{x}) - f(\mathbf{x}))^2] \right] \\ &= \mathbb{E}_{\mathbf{x}} \left[\mathbb{E}_{\mathcal{D}}[g^{(\mathcal{D})}(\mathbf{x})^2] - 2\mathbb{E}_{\mathcal{D}}[g^{(\mathcal{D})}(\mathbf{x})]f(\mathbf{x}) + f(\mathbf{x})^2 \right]\end{aligned}\quad (3.2)$$

where $\mathbb{E}_{\mathcal{D}}[g^{(\mathcal{D})}(\mathbf{x})] \equiv \bar{g}(\mathbf{x})$ is a kind of average prediction if we would repeat the prediction for many datasets of the same distribution; $f(\mathbf{x})^2$ describes the variability of the true function f .

The expected out-of-sample error is then rewritten in terms of $\bar{g}(\mathbf{x})$ (Abu-Mostafa et al., 2012):

$$\begin{aligned}\mathbb{E}_{\mathcal{D}} \left[E_{\text{out}}(g^{(\mathcal{D})}) \right] &= \\ &= \mathbb{E}_{\mathbf{x}} \left[\underbrace{\mathbb{E}_{\mathcal{D}}[g^{(\mathcal{D})}(\mathbf{x})^2] - \bar{g}(\mathbf{x})^2}_{\mathbb{E}_{\mathcal{D}}[(g^{(\mathcal{D})}(\mathbf{x}) - \bar{g}(\mathbf{x}))^2]} + \underbrace{\bar{g}(\mathbf{x})^2 - 2\bar{g}(\mathbf{x})f(\mathbf{x}) + f(\mathbf{x})^2}_{(\bar{g}(\mathbf{x}) - f(\mathbf{x}))^2} \right]\end{aligned}\quad (3.3)$$

The first term is the variance of the predictions around the average prediction. The second term is the squared bias which describes how much the average prediction $\bar{g}(\mathbf{x})$ would deviate from the true function $f(\mathbf{x})$. It is zero as soon as the model is in the hypothesis set of the model.

Figure 5 illustrates both terms by an example: The underlying true function is a sine-wave from which two samples are randomly drawn to fit the model. While the model in the left column is just $f(x) = \text{const.}$ (the mean of the two samples), the model in the right column is a linear function (the straight line through the two samples).

Generally, the linear model is a better approximation to the underlying function in this interval. Nevertheless, for only two samples, the linear model is very sensitive to the location of the points which can be seen by the large errors in the right column.

As soon as more samples would be used to determine the linear model, the variance error would decrease and the total error would be governed by the bias of the models. Since the bias of the linear model is lower than that of the constant one, it will at some point be better than the

constant model.

It can be concluded that a model may have a low bias since the average prediction may fit the data well, but the lack of samples may blow up the variance of the model's prediction when it is sensitive to variations of the training set. A model that is too simple exhibits a large bias error.

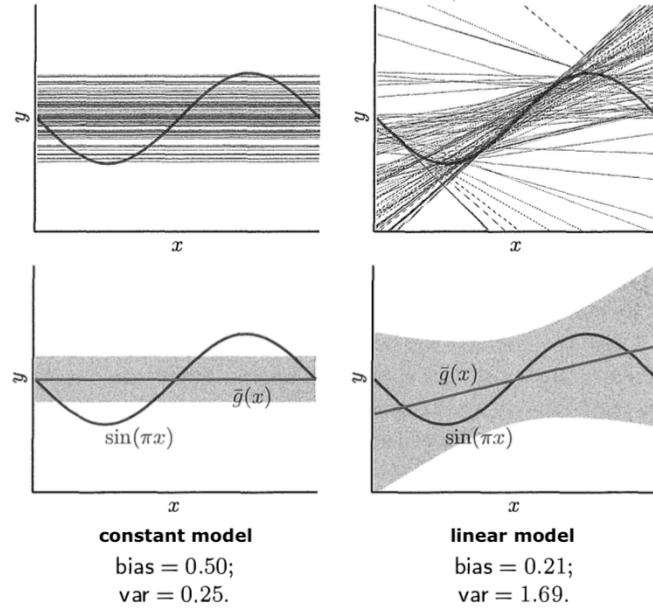


Figure 5: An illustration of the bias-variance tradeoff. Two models of different complexity are used to predict the underlying true function (a sine-wave) from which only two points are sampled for model building. The model in the left column is $f(x) = \text{const.}$ (the mean of the two points). The model in the right column is a linear function (the straight line through the two points). One can see that the prediction of the linear model is too complex for the sample size two. Taken from [Abu-Mostafa et al. \(2012\)](#).

Another visualization of the Bias-Varience tradeoff is figure 6. The graph shows how the in-sample and out-of-sample error changes for increasingly complex models. While the in-sample error decreases until the model is able to replicate

As soon as the model complexity is too large, the out-of-sample error increases, while the in-sample error continues decreasing.

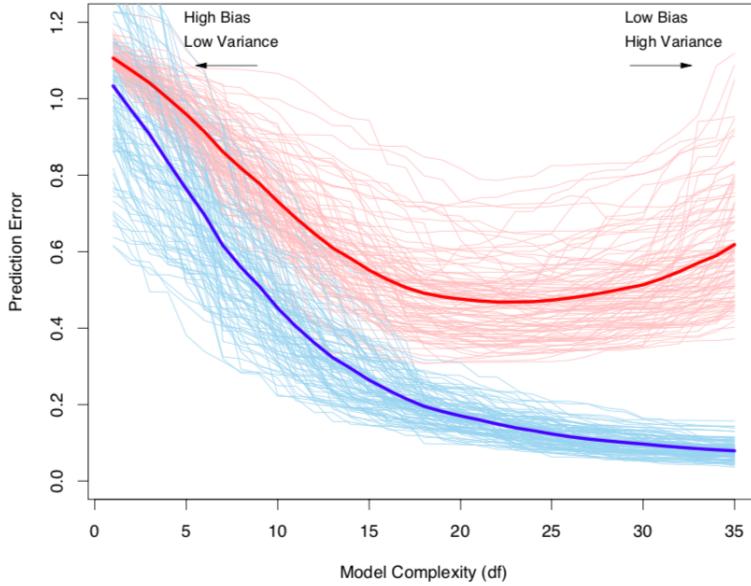


Figure 6: This graph shows an example of in-sample error (blue) and out-of-sample error (red) versus model complexity. In-sample error decreases when model complexity is increased. The out-of sample error only decreases to a certain point from which on the model 'overfits' on the data, i.e. the error increases. Taken from (Hastie et al., 2009).

Both concepts, the VC Analysis of the section above and the Bias-Variance Analysis describe the same effect but from another perspective, which is sketched in figure 7. The first defines the generalization error in terms of the difference of predictions and hindcasts (out/in-sample), where one decreases and the other increases with more samples. In the second, only the variance decreases because the bias does not depend on the dataset (equation 3.3).

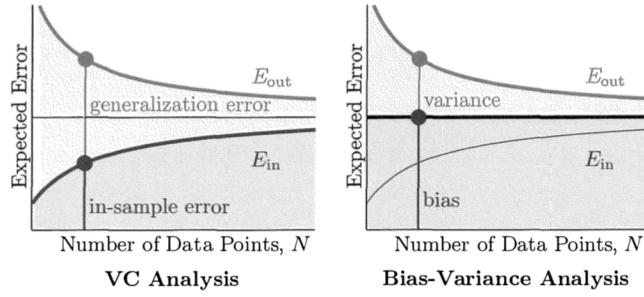


Figure 7: Comparison of concepts: Generalization error and bias-variance tradeoff. Taken from Abu-Mostafa et al. (2012).

Finally, one can see that increasing the number of samples has different effects for simple and complex models (fig. 8). While both models show decreasing errors, a complex model may need a higher number of samples to reduce its variance to a level that is comparable to the simple model whereas the simple model can not benefit from more samples because the variance term is already lower and the bias term cannot be improved by more samples.

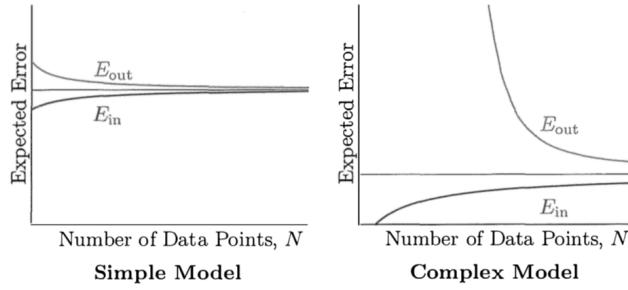


Figure 8: A complex model may need more samples until its variance is decreased, while the simple model can not improve its bias term with additional samples. Taken from [Abu-Mostafa et al. \(2012\)](#).

Regularization A technique to reduce the variance error is *regularization*. In linear regression models, the variance originates from large multiplication factors, i.e. coefficients. In a multiple linear regression the coefficients can be minimized by adding penalty terms into the cost function. Common regularizations include L1 which reduces absolute values of coefficients or L2 which penalizes squared coefficients. Yet this is not the only method to regularize a model. Especially in the field of neural networks a variety of methods exist:

Besides to penalizing coefficients (known as weight or *kernel-regularization*) one can use *dropout* ([Srivastava et al., 2014](#)), which omits nodes inside the network in each iteration with some probability, so that the result does not get too dependent on a specific node in the network. *Batch normalization* is mainly added to speed up the training but it also regularizes the model ([Ioffe and Szegedy, 2015](#)). *Early stopping* means to end the weight optimization process when the out-of-sample error stops decreasing (fig. 6).

For decision trees, *pruning* and *ensemble*-building with differently resampled training sets or different subsets of predictors serve as regularization.

3.1.2 Estimating out-of-sample performance in practice

Since Statistical Learning Theory does not provide accurate estimates how a model will perform in the future, one has to consider other ways. There is a simple, yet powerful technique that makes it easy to obtain estimates of out-of-sample performance. One technique of course is

time-splitting, where one part of the dataset is used to train and another one is used to validate, but splitting the dataset reduces the number of samples available for training. Another option that does not reduce the training set size is *cross-validation* (CV). The idea is the following: Leave the beginning 10 % of samples for validation, train on the remaining 90 %. Predict the 10 % that were left out in training and move on to the next 10 % to validate. One can also do CV by leaving out only one sample to validate ("Leave-One-Out" CV). In meteorology however, we are dealing with autocorrelated time series. If one does not account for autocorrelation, the model could show additional "artificial skill", i.e. better scores by nearby samples.

The solution to correlated samples can be blocked cross-validation (Racine, 2000) that leaves out some samples to decorrelate training and validation datasets (figure 9).

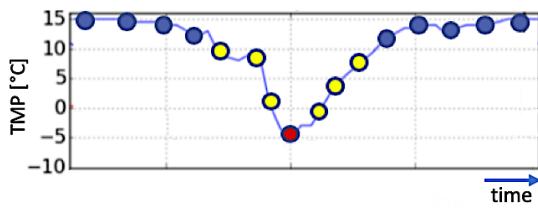


Figure 9: An illustration for how cross-validation can be done for correlated timeseries: The timeseries of temperature is partly used for training (blue points) and evaluated at one or more points (in red). It is however important to decorrelate both datasets by leaving out a few samples (yellow points).

When applying neural networks with early-stopping (see section above), one needs a validation set for training. In this work, the CV was used for this task. This also means that the neural net is somewhat optimized on the CV dataset. In the words of Abu-Mostafa et al. (2012) the validation set has become "compromised", i.e. an unbiased estimate of out-of-sample performance using the validation set is not possible anymore.

Summary The Statistical Learning Theory is able to say probabilistically, whether a specific learning task (given a number of samples and a model of some complexity) is doomed to fail, or is a feasible task. Unfortunately, the theoretical inequalities are seldomly useful outside of a textbook example. Hence, cross-validation (CV) and bootstrap resampling is used to estimate the predictive performance: CV estimates the variability of the model performance with respect to different datasets and bootstrapping is done to compute a confidence interval for the performance metrics.

3.2 Prediction methods

Physical ice accretion models The Makkonen model ([Makkonen, 1984, 2000](#)) was probably the first model to predict icing of powerlines. It takes the simplified form

$$\frac{dM}{dt} = \alpha_1 \alpha_2 \alpha_3 w v D \quad (3.4)$$

where M is the ice load [kg/m], w is the liquid water content [kg/m³], v is the horizontal wind speed [m/s] and D is the cylinder diameter [m]. α_1 to α_3 are the collision efficiency (the ratio of the particles hitting the object to the total amount of particles), the sticking efficiency (describing the ratio between the droplets sticking on the object to the total amount of droplets hitting the object) and the accretion efficiency (describing runoff due to incomplete freezing in the so called *wet growth* regime and other factors). The effect of the droplet diameter for example, is accounted for through the collision efficiency factor.

[Davis et al. \(2014b\)](#) refined the Makkonen model of static structures into the *IceBlade* model for rotating turbine blades, which is different to Makkonen's model in that it does not assume cylindrical form of the blades and accounts for the speed of the rotor through the ambient air.

A detailed study of the sensitivity of both models with respect to internal parameters, such as the assumed median volume diameter (MVD), collision or accretion efficiency factors, and externally forced boundary conditions to the model, such as temperature, liquid water content and droplet diameter, is provided in [Weiβ \(2018\)](#).

To summarize, the icing rate theoretically depends to a large degree on temperature, liquid water content, droplet sizes and wind speed. A reasonably well statistical model should parameterize these variables from the predictor variables provided to the model in order to detect significant icing rates.

Statistical and machine-learning models The main purpose of this thesis is to evaluate various statistical models for icing forecasts, which will be described one after the other in the following sections. There are many books and reviews about statistical and machine-learning models. Especially notable are [Hastie et al. \(2009\)](#), who covers a wide range of algorithms in detail and [Hsieh \(2009\)](#), who provides a review of successfully applied models in the domain of atmospheric science. As the field develops in quite a pace, more appropriate summaries may have been published since. Nevertheless, the fundamental element as introduction to machine-learning is the methodology of crossvalidation, the concept of complexity in statistical learning and its implications for choosing the model architecture.

Terminology Which model is a statistical model and which is a machine-learning model? There seems to be no common agreement to distinguish these two approaches but as Breiman et al. (2001) noted, a central part of statistical modeling is to assume a data model, i.e. the mathematical form of the *data-generating process*⁶, i.e. binomial, poisson, gaussian. Algorithmic modeling, however, does not assume anything about the data-generating process but simply consists of rules to algorithmically minimize errors given some error measure.

Statistical and machine-learning algorithms

3.2.1 Logistic regression

The (binary) logistic regression model describes the logit of the probability p of an event, $\text{logit}(p) = \log \frac{p}{1-p}$, with a linear function of the P predictor variables x_i :

$$y = \log \frac{p}{1-p} = \beta_0 + \sum_i^P \beta_i x_i \quad (3.5)$$

The logit transforms the resulting equation for p into the range [0,1] assuming a binomial error distribution.

The ordinary least squares formulation of the model aims to minimize

$$\min_{\beta_0, \beta_i} \left\{ \sum_t \sum_i (\beta_0 + \beta_i x_i^t - y^t)^2 \right\}, \quad (3.6)$$

summing over time/samples (index t) and predictors (i) or equivalently in matrix notation⁷,

$$\min_{\beta} \left\{ \|\beta \mathbf{x} - \mathbf{y}\|_2^2 \right\}. \quad (3.7)$$

When estimating the coefficients β_i from data, one has to apply *regularization* (see section 3.1.1) to avoid overfitting to reduce the variance of the prediction error and reach a stable prediction model. A straightforward solution is the Tikhonov regularization (Abu-Mostafa et al., 2012; Tikhonov, 1963) which adds the term $\|\beta\|_2^2$ to the cost function, eqn. 3.7, yielding⁸

$$\min_{\beta} \left\{ \|\beta \mathbf{x} - \mathbf{y}\|_2^2 + \lambda \|\beta\|_2^2 \right\}. \quad (3.8)$$

with λ being a free parameter determining the strength of regularization between $\lambda = 0$, the

⁶Covered by Wilks (2011), Ch. 6.3

⁷adding β_0 and 1 as first elements into the vectors β and \mathbf{x} respectively; the summation over time is dropped for simplicity

⁸assuming normalized predictors x_i ; otherwise we would have to divide by the variance of each predictor in the added term for the regularization to be fair

unconstrained ordinary least-squares regression, and $\lambda \rightarrow \infty$, in which case all coefficients β tend to zero. Setting all coefficients except β_0 to zero results in the simplest model, i.e. plain bias-correction, which just predicts the mean of the target variable. Thus, increasing λ decreases the model complexity.

Experience shows that if the number of predictors is too large, the coefficients become larger and partially compensate each other to achieve the lowest possible sum of squares. Evaluated on independent data, this will lead to an extremely high error compared to in-sample error as it overfits the data. Yet, there are multiple ways to deal with this: [Wilks \(2011\)](#) recommends *forward selection*, while in the context of machine-learning *shrinkage* methods ([Hastie et al., 2009](#)) are more popular.

Especially LASSO (least absolute shrinkage and selection operator) with its L1 regularization term $\|\beta\|_1^2$ is used quite commonly, as it is able to set coefficients exactly to zero ([Hastie et al., 2009](#)), thus effectively doing feature selection. [Hastie et al. \(2017\)](#) discusses the question of which method is best. The present work uses L2 regularization.

3.2.2 Generalized additive models

A way to generalize multiple linear regression models, where the predictand y is modeled as a linear function of p predictor variables x_i ,

$$y = \beta_0 + \sum_i^P \beta_i x_i, \quad (3.9)$$

to also include non-linear dependencies on features, is to replace each linear function of x_i , (i.e. $\beta_i x_i$), by some arbitrary function of x_i , i.e. $f_i(x_i)$:

$$y = \beta_0 + \sum_i^P f_i(x_i) \quad (3.10)$$

A model of this kind is called generalized additive model ([Hastie and Tibshirani, 1990](#)). The model prediction y depends additively on all predictors, thus its name. This property allows us to investigate the effect of each predictor independently of the rest, which is not the case for neural networks and the most other machine learning models that use nonlinear functions of predictors.

To predict the probability p that $y = 1$, one has to use a logistic link function $g(\cdot)$ on y :

$$p(y = 1) = g(y) = \exp(y)/(1 + \exp(y)) \quad (3.11)$$

In the present work, this logit link function and the binomial distribution is used for all predictions since the predictand is always a probability. This combination is called logistic GAM.

Spline terms To model the target's dependence on the predictor variables, regression splines are used within this work. The spline function $f(x)$ for one predictor x is constructed as the sum of shorter wavelength signals, so called B-splines $B_{l,q}(x)$:

$$f(x) = \sum_{l=1}^K \beta_l B_{l,q}(x) \quad (3.12)$$

This is illustrated in figure 10. The actual implementation by [Servén and Brummitt \(2018\)](#) imposes a penalty on the second derivative of the fitted curve to enforce smoothness and thus reduce the risk of overfitting, a common method for spline-fitting that dates back at least to the survey of [Reinsch \(1967\)](#).

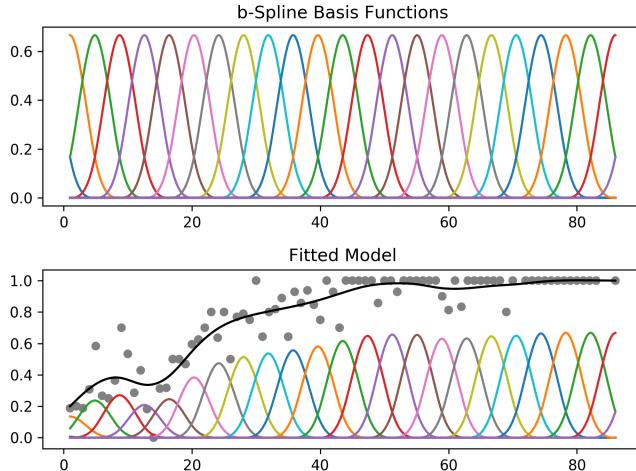


Figure 10: The upper part shows multiple b-spline basis functions over some interval. The lower part visualizes the contribution of each basis function on the resulting spline. Created by [Servén and Brummitt \(2018\)](#).

3.2.3 Artificial neural networks

The most basic neural network may be the fully connected network (FCN, fig. 11) which is in fact equivalent to multiple linear regression equations where each output may be mapped to the

range 0-1 by an activation function f . So the prediction y of the output node j is given by

$$y_j = f \left(\sum_i w_{ji} x_i + b_j \right) \quad (3.13)$$

where w_{ji} is the weight for the predictor variable x_i and b_j is a constant additive term called bias. Common activation functions include the sigmoid function, ramp functions like RELU or the identity function ($f(x) = x$).

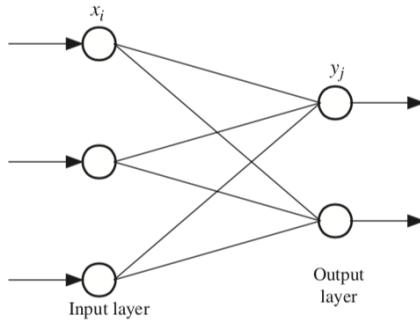


Figure 11: The fully connected network or *perceptron* model consists of a layer of input neurons connected directly to a layer of output neurons.

The present work uses 'ordinary' neural networks, also known as *multi-layer perceptrons* (MLP, fig. 12). Such a model consists of a chain of FCNs, where the output of one FCN is taken as input for another FCN. Layers of nodes whose outputs are used as inputs for another layer are called *hidden layers*. In theory, only a single hidden layer with a finite number of nodes is enough to approximate any continuous function⁹.

Back-propagation What makes the artificial neural net (ANN) extraordinary is how the optimal weights w_{ji} on each level of the ANN are found, as there may be thousands of them and each of them depends on many others. The algorithm is called back-propagation and makes use of the chain-rule of differentiation to minimize the objective function¹⁰ J . At first, the gradient of J with respect to every weight is calculated by backward propagation of the training error and in the second part the weights are iteratively changed towards the minimum of J .

Architecture Géron (2017) provides some advice on how to choose the architecture but as there is no single best architecture, one has to optimize it for every new dataset/task.

⁹This is called the Universal approximation theorem by Cybenko (1989).

¹⁰ J is also called error function, cost function or loss function.

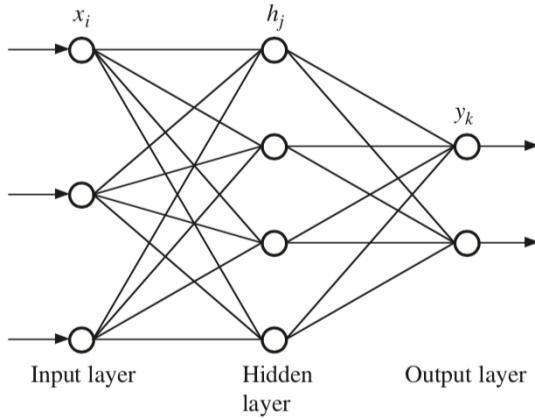


Figure 12: The multi-layer perceptron (MLP) model with one hidden layer of neurons between the input layer and the output layer.

Finally, another useful technique that has been introduced in the last years is *embedding*, applied for example in Rasp and Lerch (2018). The technique consists of adding time-constant predictors for every weather station, e.g. altitude of station, altitude of model gridpoint, for a model predicting all stations, which allows to use the same model for different stations while making use of the similarities.

3.2.4 Classification and regression trees

Tree-based methods recursively split the feature space into rectangular partitions such that samples with the same outcome are grouped together. A common tree-based method is 'Classification and Regression Trees' (CART), which predicts the majority outcome of the samples in each region or $f(x) = \text{const.}$ for regression.

The algorithm To illustrate the algorithm for classification, suppose there are only two predictor variables x_1 and x_2 for the binary predictand y . If the outcome depends on x_1 , then there will be made a partition at point x'_1 . In both branches left and right of the partition point ($x_1 < x'_1$ and $x_1 \geq x'_1$), the prediction of the decision tree is given by the majority of outcomes y in that branch. The best partition point is the one which minimizes classification error. The same procedure is followed to split along the second direction x_2 . As soon as both splits have been determined, one can compare the resulting classification error and choose the one which minimizes the prediction error.

This process is then repeated for each partition until a given depth of a tree is reached. Clearly, a tree will give zero in-sample error at some depth, as it can continue until only one outcome is

left in each branch. To find a tree with optimal out-of-sample error, [Hsieh \(2009\)](#) recommends to "grow the tree to a large size" and remove nodes sequentially depending on which node is least important, i.e. increases the error the least. From this sequence of trees one may choose the tree with the smallest regularization function $J(L) = E(L) + PL$, where L is the number of leaf nodes and P is a weight penalty, which can be determined by cross-validation. The in-sample error E of the classification is given by the gini impurity

$$E(L) = \sum_{l=1}^L \sum_{k=1}^K p_{lk}(1 - p_{lk}) \quad (3.14)$$

where p is the proportion (btw. 0 and 1) of the data in region l belonging to class k . This formula penalizes large inhomogenities, i.e. errors, in the $L \cdot K$ partitions.

Ensemble methods A decision tree is probably the most transparent prediction model, since the reason for a prediction $y(x)$ is completely determined by a sequence of binary decisions on the values of its predictor variables. However, as [Breiman \(1996\)](#) noticed, this type of model lacks prediction performance in some cases, especially when the prediction performance varies a lot for different datasets (for example for data from one year versus another). In these cases he suggested to train several trees on similar datasets, which are constructed from an original dataset by bootstrap resampling. In the prediction step, one averages all the independent predictions (in regression) or follows the majority of the predicted class (in classification). This method is called 'bootstrap aggregation' or *bagging* for short. A common modification to bagging is *random forests*, for which one uses a random subset of all predictors for every node ([Breiman et al., 2001](#)). A separate idea is *boosting* ([Freund et al., 1996](#)), where trees are trained sequentially on the same training set but with different weights applied to samples which are responsible for a large prediction error. Those are weighted to be more important than other samples which were already predicted better. In this way, each new tree will improve the prediction where it previously was bad. Apart from the methods described here, there are many modifications and similar methods like C4.5, CHAID and MARS.

3.2.5 Support vector machines

The support vector classification (SVC) algorithm was developed in the early 1960's to separate classes of patterns without the use of modern electronic computers. Since the inventors were able to come up with models that made no errors on the training set, they asked themselves why there is hope to achieve low error out-of-sample too ([Vovk et al., 2015](#)). The study of this question lead to the development of statistical learning theory (section 3.1).

The construction of the algorithm will be given in three steps, following Hsieh (2009): At first, the algorithm is constructed for the simplest case, where points are linearly separable (without classification errors). Then one modifies the method to allow for classification errors and finally the method is extended to nonlinear classification.

Intuition An example for a linearly separable case is given in figure 14, where one can draw a straight line in such a way that all positive samples lie on one side and all negative samples on the other side. The main idea of the SVM algorithm is, that the line which exhibits the largest distance to any of the sample points is also the most probable decision boundary ("maximum margin classifier"). This is intuitive since the line ensures the lowest error rates if noise is added to the samples' positions for example. As linear-separability is only a special case, one can account for classification errors by adding a penalty term into the cost function.

Lastly, nonlinear classification (e.g. by a non-straight line in two-dimensional space) can be introduced by a transformation function that is applied to the input dimensions, for example by reprojecting the two-dimensional input space (x, y) by two predictor variables onto a disc of radius r by $r^2 = x^2 + y^2$, resulting in a space where the samples' class y (either +1 or -1) is a function of r only: $y = y(r)$. A linear classification in this space (a number r^* , separating smaller and larger values of r) yields a non-linear classification in the input space: a disc with radius r , that separates samples inside the disc from samples outside the disc.

Algorithm construction At first, a function $y(\mathbf{x})$ is defined, whose sign indicates of which class a given point \mathbf{x} is. For one class, $y > 0$, whereas for the second class $y < 0$. The "line" where $y(\mathbf{x}) = 0$ is called decision boundary.

The function for the decision boundary is given by

$$y(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + w_o = 0 \quad (3.15)$$

With two predictors, the situation is easily visualized in three dimensions. In this case, $y(\mathbf{x})$ can be seen as the height of a plane above the horizontal plane $y = 0$. The plane intersects the horizontal plane where $y(\mathbf{x}) = 0$. The intersection of two planes is a line - the decision boundary that separates the two classes, i.e. the two regions on the horizontal plane spanned by x_1 and x_2 . Figure 13 gives a top-down view on this example.

One main characteristic of the SVC is its margin, visualized in figure 14. The margin l is the shortest distance from the decision boundary to any of the training samples (lowest norm in the feature space). The normal distance from the decision boundary to some point \mathbf{x} is given by its

projection to the unit vector of the decision boundary \hat{w} , while x_0 lies on the boundary:

$$\hat{w} \cdot (x - x_0) = \frac{\mathbf{w} \cdot (x - x_0)}{\|\mathbf{w}\|} = \frac{\mathbf{w} \cdot x + w_0}{\|\mathbf{w}\|} = \frac{y(x)}{\|\mathbf{w}\|} \quad (3.16)$$

The unit vector of some initial vector w is $w/\|w\|$.

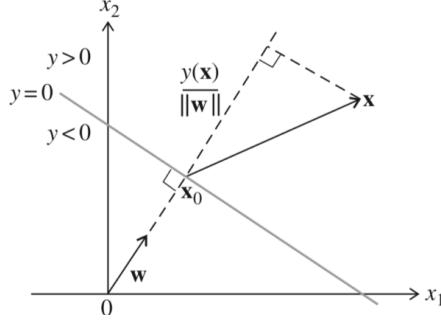


Figure 13: The basic geometrical setup: To formulate the solution in mathematical terms, one needs an expression for a point x in this two-dimensional space. The class label y (either -1 or +1) is given as function of x_1 and x_2 . The separating hyperplane, defined by $y(x_1, x_2) = 0$, needs to be found. From Hsieh (2009).

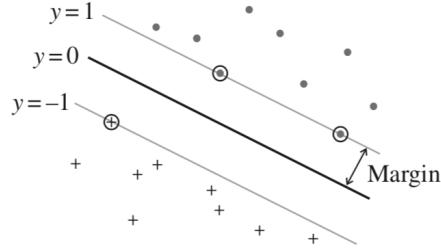


Figure 14: The support-vector classification maximizes the margin between the classes. From Hsieh (2009).

The optimization problem is then to maximize the distance to point x_n , where y_{dn} is either +1 or -1 to ensure non-negative distances:

$$\max_{\mathbf{w}, w_0} l \quad \text{subject to} \quad \frac{y_{dn}(\mathbf{w} \cdot \mathbf{x}_n + w_0)}{\|\mathbf{w}\|} \geq l, \quad (n = 1, \dots, N) \quad (3.17)$$

As scaling w by an arbitrary factor does not change the decision boundary, one may define w to be l^{-1} . This transforms the problem above into a common mathematical optimization of a quadratic function under side conditions, which is referred to as *quadratic programming* problem:

$$\min_{\mathbf{w}, w_0} \|\mathbf{w}\|^2 \quad \text{subject to} \quad \frac{y_{dn}(\mathbf{w}^T \mathbf{x}_n + w_0)}{\|\mathbf{w}\|} \geq 1, \quad (n = 1, \dots, N) \quad (3.18)$$

The above problem is solved by the Karush, Kuhn and Tucker (KKT) formula with the Lagrangian L and lagrange multipliers λ_n :

$$L(\boldsymbol{\lambda}) = \sum_{n=1}^N \lambda_n - 1/2 \sum_{n=1}^N \sum_{j=1}^N \lambda_n \lambda_j y_{dn} y_{dj} \mathbf{x}_n \cdot \mathbf{x}_j \quad (3.19)$$

In this formula, the \mathbf{x} never appears 'alone' but only in dot products of each other. This is a significant advantage in order to generalize the SVM classifier to nonlinear classification by replacing the dot product with a *kernel* function $\phi(\mathbf{x}_i, \mathbf{x}_j)$.

In most cases, it is necessary to allow classification errors because having no misclassifications means overfitting. The classification error ξ is defined to be zero if the point lies on the margin or in the correct halfspace around the decision boundary and $|y_{dn} - y(\mathbf{x}_n)|$ otherwise (fig. 15).

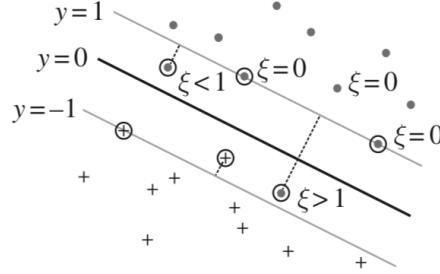


Figure 15: Support-vector classification: Non-separable case. From [Hsieh \(2009\)](#).

Finally, the resulting cost function is ([Hastie et al., 2009](#))

$$\min_{\mathbf{w}, w_0} \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum_{i=1}^N \xi_i \quad (3.20)$$

$$\text{subject to} \quad \xi_i \geq 0, \quad y_i(\mathbf{w} \cdot \phi(\mathbf{x}_i)) \geq 1 - \xi_i \quad i = 1, \dots, N$$

which maximizes the margin and contains a regularization towards simpler decision boundaries.

3.3 Verification

This section describes how the forecast quality was quantified. Following the results of section 3.1, the verification was done in the following steps:

1. **The hindcasts were produced with blocked cross-validation** (see 3.1.2).

This step should provide unbiased estimates of out-of-sample performance through multiple model evaluations with decorrelated training and validation datasets.

2. **Resampling the forecasts with blocked bootstrapping 200 times.**

To estimate the variability of a certain metric including its confidence interval, blocked bootstrapping is applied to the cross-validation predictions to replicate the correlation in the data ([Wilks, 1997](#)).

3. **Compute the probabilistic metrics** and evaluate the uncertainty over bootstraps.

4. For a range of probability thresholds (e.g. from one to 99 %), **binarize the forecasted probabilities** to 0/1 with the respective probability threshold, **compute the deterministic metrics** and evaluate the uncertainty over the bootstraps.

Note that the error rates of many ML models are not reproducible if they are run only once. This is either due to randomly initialized parameters or a stochastic optimization algorithm. To achieve reproducible results, we apply cross-validation. This also introduces variability in the training dataset as different parts of the whole dataset are used for training. To estimate the effect of stochasticity in the model optimization, one could use identical training data over and over again. This individual effect however, is not relevant to the added value of the methods. To estimate the total effect of variability, the range from the 5th to the 95th percentile of a scores is reported as the confidence interval.

Measures of forecast quality The forecast quality is assessed by the following scalar attributes ([Wilks, 2011](#)):

- Resolution, which describes the ability of the forecast to discriminate between icing and non-icing (quantified by the area under the ROC curve in this work), as well as
- Accuracy, which measures the difference between forecasted probabilities and observed values (0/1) (quantified by the Brier skill score in this work).

3.3.1 ROC & Reliability diagram

ROC diagram The ROC (receiver-operating-characteristic) diagram from signal-detection theory describes how well a binary forecast can differentiate between events and non-events.

It does so by comparing the hit rate (probability of detection, POD) to the false alarm rate (probability of false detection, POFD). Both estimates of probabilities are defined with respect to the number of observed events and non-events respectively.

Deterministic forecasts produce one single point with coordinates (POD, POFD) in the ROC diagram. Probability forecasts induce a set of points in the ROC diagram which form the ROC curve, since any probability threshold results in a different POD and POFD. In this way, the ROC diagram illustrates the forecast skill for different probability thresholds. Decreasing the threshold leads to more positive forecasts, so that in most cases some of the previously missed events will now be detected and some of the correct "no"-forecasts will become false alarms.

To determine the overall skill, the area under the ROC curve, abbreviated AUROC or AUC, can be calculated. Its value varies between AUC=0.5 for random forecasts and AUC=1 for perfect forecasts.

The ROC diagram does not show whether the forecasted probabilities p_{fc} are well calibrated, as only the ordering matters: samples with higher p_{fc} must occur more often than samples with lower p_{fc} so that the optimal threshold p^* discriminates them to maximize ratio of hits to false alarms.

Reliability diagram The reliability diagram shows how well the forecasted probabilities match the true conditional probability, e.g. consider all forecasts with probability in the range 90-100 % - how often did the event occur in these cases? Common types of error are: Overconfidence, such that for high forecast probabilities the event occurred significantly less often than forecasted, and for low forecast probabilities it occurred much more often than forecasted; Overforecasting, such that the forecasted probabilities are generally too high.

Finally, the distribution of forecasted probabilities may indicate low confidence by mostly forecasting the climatological frequency and seldomly forecasting higher or lower probabilities.

3.3.2 Brier score

The Brier score (BS) measures the distance between forecasted probabilities f_i in the range 0-1 and observed values o_i (either 0 or 1).

$$BS = \frac{1}{N} \sum_i^N (f_i - o_i)^2 \quad (3.21)$$

The perfect score of zero is achieved if the forecasted probability matches the observed value for all samples i . For better comprehensibility, we use the skill score version, where the score is normalized by the Brier score of a constant probability forecast that is the climatological

probability (BS_{clim}):

$$BSS = 1 - \frac{BS}{BS_{clim}} \quad (3.22)$$

Stanski et al. (1989) list characteristics of the BSS: One important implication for the present work is that the climatological Brier score tends to be, by chance, already quite low and thus, hard to beat by the forecast. The result is a negative BSS as can be seen for the rare icing class in the results section.

3.3.3 Economic value score

The economic value score (Richardson, 2000) of a forecast is defined as the ratio of possible economic savings by using this forecast compared to using no forecast at all.

More specifically, the value score (V) is the ratio of two expense (E)-savings: The nominator describes the savings from using our probability forecast compared to the climatological probability (as forecast). The denominator compares the previous savings to what the perfect forecast would save:

$$V = \frac{E_{clim} - E_{forecast}}{E_{clim} - E_{perfect}} \quad (3.23)$$

Specifying quantitative values for the expenses involves assigning actual expense to the possible outcomes of the forecast. The one used most commonly is a static cost-loss model that assumes an expense (in a monetary unit) of C cost for positive forecasts (we take measures), 0 cost for true negatives (nothing bad happens) and L for false negatives (the event happened but we did not prepare for it). This is formalized in the following equation with a , b and c being the number of hits, false alarms and misses (true positives, false positives and false negatives), while n is the number of total forecasts:

$$E_{forecast} = \frac{a}{n}C + \frac{b}{n}C + \frac{c}{n}L \quad (3.24)$$

In terms of common metrics like hit rate H , false alarm rate F and base rate s , this is equivalent to

$$E_{forecast} = F(1 - s)C - Hs(L - C) + sL \quad (3.25)$$

The perfect forecast has zero misses and false alarms and results therefore in costs of occurrence

rate times C :

$$E_{\text{perfect}} = sC \quad (3.26)$$

If we only know the climatological probability, our optimal decision can either be to take measures every day (at cost C) or to never take measures and lose L with probability s . Thus, depending on the ratio of cost C to loss L , there is a different economic optimum.

$$E_{\text{clim}} = \begin{cases} C & \text{if } C/L < s \\ sL & \text{if } C/L > s \end{cases}$$

This motivates the introduction of the cost-loss ratio $\alpha \equiv C/L$, which controls much of the potential value of forecasts. As [Jolliffe and Stephenson \(2012\)](#) show, is the maximum value present when the climatological frequency of the event equals the cost-loss ratio ($\alpha = s$).

Finally, after substituting all terms in equation 3.23, the economic value can be computed as

$$V = \frac{\min(\alpha, s) - F(1-s)\alpha + Hs(1-\alpha) - s}{\min(\alpha, s) - s\alpha} \quad (3.27)$$

The PEV diagram In the previous paragraphs, we saw that the potential economic value (PEV) of a forecast depends on the user-specific cost-loss ratio α . Therefore, it is not possible to measure the value in one single metric for all possible users. Instead, we can graph the PEV in a diagram against α (fig. 16).

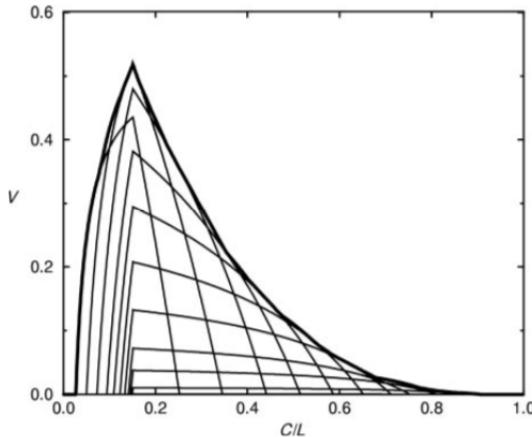


Figure 16: The PEV curve depicts the potential economic value against the cost-loss ratio. For probability forecasts it is constructed as the envelope curve using all probability thresholds. Taken from [Jolliffe and Stephenson \(2012\)](#).

Now, since equation 3.27 only describes the value of a deterministic binary forecast, what is the value of a probability forecast? We can certainly estimate the value by generating binary forecasts for many probability thresholds between 1% and 99% and then evaluate which is best. The same can be achieved visually by plotting a curve for each probability threshold and then highlighting the envelope as potential economic value like it is done in figure 16. Potential, since the maximum economic value is only realized if the optimal probability threshold is used.

The PEV curve typically looks like a pointed arch that reaches its maximum value at $\alpha = s$ where $V = H - F$ (Jolliffe and Stephenson, 2012). On both sides of the peak, the line shows a characteristic behaviour and then drops below zero sooner or later (usually only the positive values of the PEV-curve are plotted). Can we understand what shapes the curve on both sides of the peak? Why do different probability thresholds result in these left or right-skewed pointed arches of figure 16?

Each curve represents a different threshold probability. If one wants to capture more events (increase the hit rate), one can reduce the threshold. This has two effects (see figure 17):

1. Some forecasts that turned out to be misses (false negatives), will now be classified as hits (true positives), since the forecasted probability (that does not change) will at some point be higher than the threshold probability (that is reduced).
2. On the other hand, some forecasts that were correct rejections will now become false alarms for the same reason as in point 1.

		observed?	
		yes	no
forecasted?	yes	TP (C)	FP (C)
	no	FN (L)	TN (0)

Figure 17: The expense matrix assigns cost C to true positive (TP) and false positive (FP) forecasts, loss L to false negative (FN) forecasts and zero expense to true negatives (TN). Increasing the probability threshold converts false negatives to true positives but also true negatives into false positives.

The interesting point here is that both effects are not equivalent in what they contribute to economic value. The conversion of misses to hits (1) reduces the expense from L to C , while the conversion from true negative to false positive (2) increases the cost from 0 to C . Looking back to the PEV diagram, one will notice that the left part of the diagram where $\alpha \ll 1$ (i.e.

$C \ll L$) is associated with substantial savings from the reduction of misses (1) since $L - C \gg C$. On the rightmost side, where $L \approx C$, effect 2 is responsible for saving costs by reducing false alarms while additional misses are not that economically relevant.

To see the impact of lowering the threshold, we calculate the difference in (relative) economic value (formula 3.27) by increasing a by one and decreasing c by one (effect 1) as well as increasing b by one and decreasing d by one (effect 2). The ratio of both effects is $(1 - \alpha)/\alpha$ for $\alpha < 1/2$ and $\alpha/(1 - \alpha)$ else (see appendix A). The formula simply states

- how many more false alarms one should allow in order to gain one additional hit in the case of $\alpha < s$, where false alarms are cheap and misses are expensive and
- how many hits we are willing to loose to avoid one false alarm in the case of $\alpha > s$, where false alarms are expensive and misses less important.

These results are consistent with the extreme case of $\alpha = 0$, where one should always take measures, equivalent to reaching a hit rate of 1, and $\alpha = 1$, where one should never take measures, equivalent to reducing the false alarms completely, i.e. never issuing a positive forecast.

Finally, one can notice a result that may seem contradicting:

- When $\alpha = s$, one percent increase in hit rate has the same effect on economic value as one percent lower false alarm rate (when equation A.7 equals minus A.8).
- On the other hand, the effect of one additional hit is the same as one less false alarm only if $\alpha = 1/2$.

Both results do not contradict each other, since one percent false alarm rate equals a higher number of false alarms when s is low, because one divides by the number of non-events.

Consequently, if $\alpha \neq s$, an increase in CSI (critical success index, Wilks (2011)) may not necessarily increase economic value, since in economic value, one percent lower false alarm rate may be more important than a two percent higher hit rate (fig. 18).

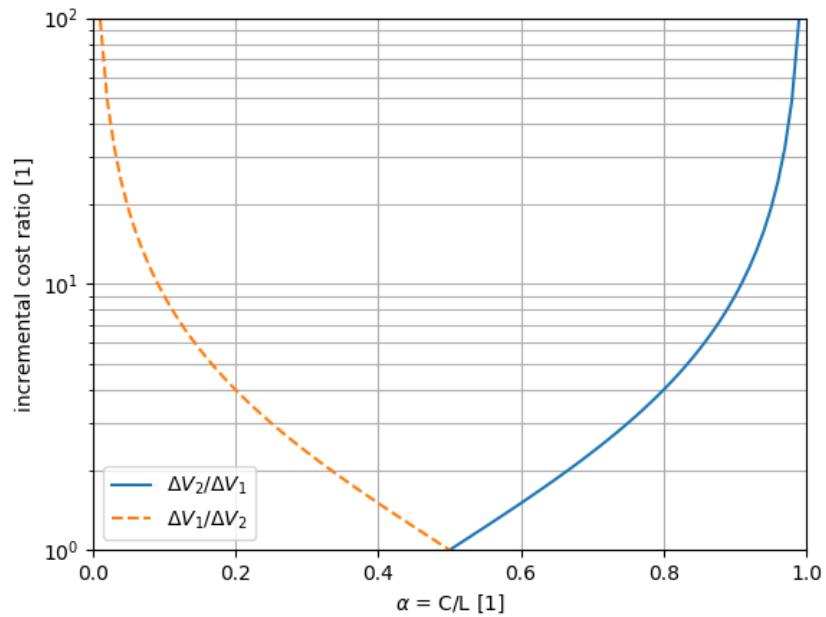


Figure 18: A variation in the probability threshold changes the (relative) economic value by changing misses to hits and correct rejections to false alarms and vice versa. The orange dashed line describes how many additional false alarms are acceptable for one additional hit while not lowering the economic value ($\alpha < 1/2$). The blue line shows how many hits can be lost without lowering the economic value ($\alpha > 1/2$) for one additional correct rejection respectively. This quantifies the tradeoff indicated in fig. 17.

4 Results

Forecasts of icing can be issued in binary form, representing icing (1) and non-icing (0), or in a probabilistic form, taking on all values from zero to one. As the input for the icing forecast are continuous variables from direct model output (DMO) of NWP models, the icing model can be seen as the mapping $f(x_1, \dots, x_n) = y$ with predictor variables $x_i \in \mathbb{R}$ onto the binary target $y \in \{0, 1\}$.

The forecast performance is verified by cross-validation (see section 3.3) to observations of three binary types of icing (see section 2.1):

- meteorological icing,
- instrumental icing and
- visible accretion.

Note that the meteorological icing classification from observation data also involves a binary classification, as it binarizes measured continuous variables according to predefined thresholds on temperature and spread (temperature $< 0^\circ\text{C}$ and dewpoint spread $< 2\text{ K}$).

In the following subsection, various performance scores are presented for different model configurations. While POD, POFD, and AUROC only check the resolution attribute of the forecast, that is the ability of the forecast to discriminate between icing and non-icing, the BSS verifies the accuracy and calibration of the forecasted probabilities, i.e. the difference between forecasted probabilities and observed values.

To achieve the highest possible AUROC, the predicted probabilities do not have to be calibrated but the ordering of forecasted probabilities $p(\text{icing})$ must be such that for some probability threshold $p^* \in (0, 1)$ the icing event only happens if $p(\text{icing}) > p^*$. For example, a model may be very underconfident, i.e. predict probabilities near 50 % most of the time. If we now verify that icing occurs only when the predicted probability is above 50 %, then the model has a high AUROC, even if there is never any icing at 45 % predicted probability. The same is the case if the predicted probabilities are biased, for example if the predictions are around 90-100 %. If there is a threshold probability that separates the two cases, the model is fine in terms of AUROC.

However, to reach a high BSS, each value of the predicted probability is important, because the predicted probabilities have to be as close as possible to the observed frequency. Assume that there was an icing event: A predicted probability of 80 % is 4 times as bad as one of 90 % since differences are squared in the Brier score.

Results outline In all sections below, the goal is basically to find the best model to discriminate icing from no-icing of the various event classes. In the first section we use an empirical-rule

on temperature and dewpoint spread to predict icing. This model, that builds on meteorological background knowledge, serves as baseline model that uses no observational data at all (since there is no training process), so that we can quantify the added value of all following *data models*.

In the following sections, the model complexity is increased from Logistic Regression over Generalized Additive Models to Artificial Neural Nets: While Logistic Regression produces calibrated probabilities, it is not able to model nonlinear effects in the predictor variables. The Generalized Additive Model, however, has the ability to do that. The next step is allowing the effect of one predictor to depend on the value of another predictor. Neural nets have this feature. Finally, there are models that work significantly different compared to the earlier ones, but are worth evaluating their results: Gradient-boosting models (a decision-tree model) and Support Vector Classification models.

Unless stated otherwise, the NWP data is taken from 780 m MSL at the nearest grid point to the site. The scores are generated by verification of all forecast hours, from one to 72 hours. Confidence intervals have been generated as described in section 3.3 and describe the range between the 5th and 95th percentile of the distribution of a score. The calculations used 10 variables: Temperature, dewpoint spread, wind speed and direction, cloud-base height, ice load and rate of the Makkonen and the IceBlade model (initialized with the values of the previous forecast with lead time +24h) as well as the time-of-day (0-23).

4.1 Empirical icing classification of direct model output

The simplest possible icing model results from classifying an atmospheric state as either icing or no-icing conditions. This can be achieved by simply thresholding DMO temperature and dewpoint spread with their respective critical values. The decision rule which has been applied in this work is based on meteorological background knowledge. It assumes that ice accretion is only possible for sub-zero ($^{\circ}\text{C}$) temperatures and moisture-saturated air. As we know that numerical models' variables are grid-box mean values and hardly reach 100 % RH, we set the spread threshold to 2 K to capture also events where the grid-box mean RH would not support full saturation. The resulting conditions with probable icing are then classified as icing periods. The opposite cases, where we expect no icing to occur, are classified as non-icing periods.

This simple classification of DMO is then used to issue binary forecasts for icing. As we use the same criteria for all types of icing (section 2.1), we obtain three different verification results, one for each type. Figure 19 shows the detection and false alarm rates for the three event categories. All results are summarized in table 3.

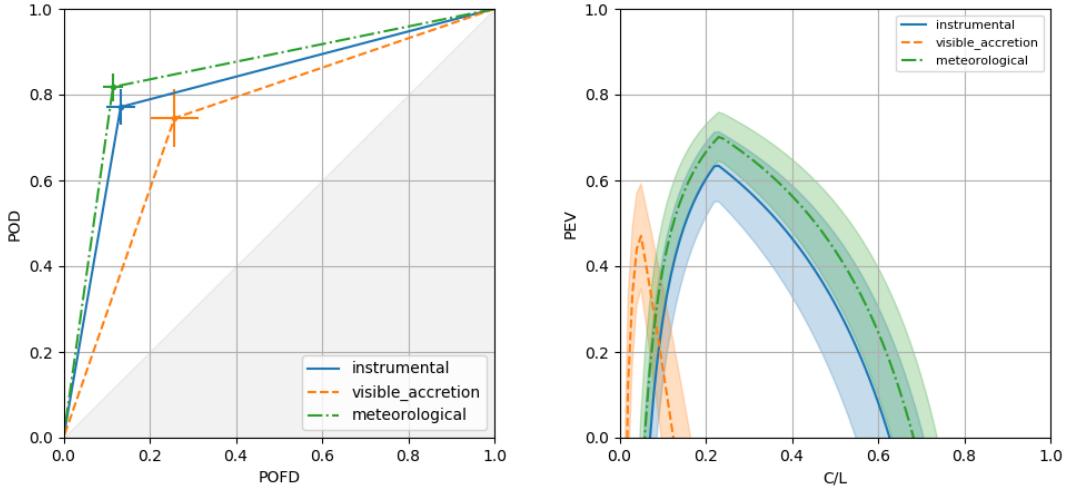


Figure 19: Verification of the empirical icing classification on NWP data to predict meteorological and instrumental icing as well as *visible accretion* periods using the ROC diagram (left) and the PEV diagram (right).

The empirical icing classification can already discriminate icing from no-icing as measured by the AUC for the case of meteorological and instrumental icing (0.83-0.86). Even for the very rare (4.7 %) class of visible icing, the AUC is still above 0.75 but is associated with a considerable error bar, indicating that there are effects that are not reliably captured by this classification which thus works sometimes better, sometimes worse for this class. The ROC diagram, figure 19, confirms that all three cases were clearly more accurate than random forecasts but that there is still room for improvement, especially for the visible accretion phase.

If we interpret the binary non-probabilistic predictions as probabilities, we can also calculate the calibration with the Brier skill score, knowing that the score will be rather low as the forecast by the empirical classification is either zero or one. As expected, the BSS is worse than climatology for meteorological and instrumental icing and only slightly more valuable than climatology for visible accretion.

Type of icing	Model	Sample size	Baserate obs.	Baserate fc.	BSS_{clim}	AUC
Meteorological	Identity	19949	0.232	0.261	0.356	0.858
Instrumental	Identity	19561	0.224	0.256	0.22	0.827
Visible accretion	Identity	19583	0.047	0.256	-4.137	0.762

Table 3: Summary of verification results for the empirical classification verified against three types of icing.

Figure 19 shows the PEV curves that result from the ROC curve in figure 19. While the forecast is of significant value for a wide range of C/L ratios for meteorological and instrumental icing, this is not the case for visible icing, where the forecasts are valuable only in a very narrow C/L band below 0.15 C/L.

4.2 Physical icing models and logistic regression

In the last section, an empirical classification rule on temperature and dewpoint spread was used to obtain binary icing forecasts from NWP variables. Now, the binary forecasts shall be obtained by logistic regression of NWP data. Contrary to the previous section however, all NWP variables are taken into account in this regression.

The result of the regression with all NWP variables is then compared to the physical modeling approach, which is performed by the Makkonen model introduced in section 3.4. It uses NWP data as boundary conditions to directly predict the evolution of ice load on the turbine. However, before being able to verify its forecasts, a threshold on the ice load has to be chosen to obtain a binary forecast, since the mass of ice is a continuous variable.

Different thresholds yield different scores through more or less sensitive forecasts. To obtain scores that are general in that they do not depend on a single threshold, all thresholds are evaluated at once by transforming the ice load into the probability range (0-1) for ice-loads between 10^{-5} and 10^{-2} kg/m by the following formula: $(\log_{10}(\text{ice load}) + 5)/3$. In the subsequent ROC analysis, the score is evaluated using a wide range of probability thresholds and thus ice load thresholds, since the predicted probability was just defined to be proportional to ice load. Nevertheless, as these values are not 'true' probabilities, one can not expect them to be calibrated.

Predicting instrumental icing Instrumental icing is present when ice is visually detected by the camera on the turbine. This should be the case when the predicted ice load by the Makkonen model exceeds a certain threshold. In the present study, a plausible range of thresholds (10^{-5} to 10^{-2} kg/m) was taken from [Strauss et al. \(2019\)](#) who also investigate how the scores change for different thresholds of ice load. Figure 20 shows that the prescribed transformation function for ice load produced better discrimination than logistic regression of NWP variables. The PEV diagram on the upper right shows significantly higher potential economic value for C/L between 15 and 50 %. The logistic regression is of similar PEV only for very low cost-loss ratios due to a higher detection rate for lower probability thresholds. The transformed ice load from the Makkonen model indeed shows a lack of calibration, but its calibration curve also suffers from a too small sample size ([Stanski et al., 1989](#)) as most forecasts turned out to be either zero or 100 % probability forecasts.

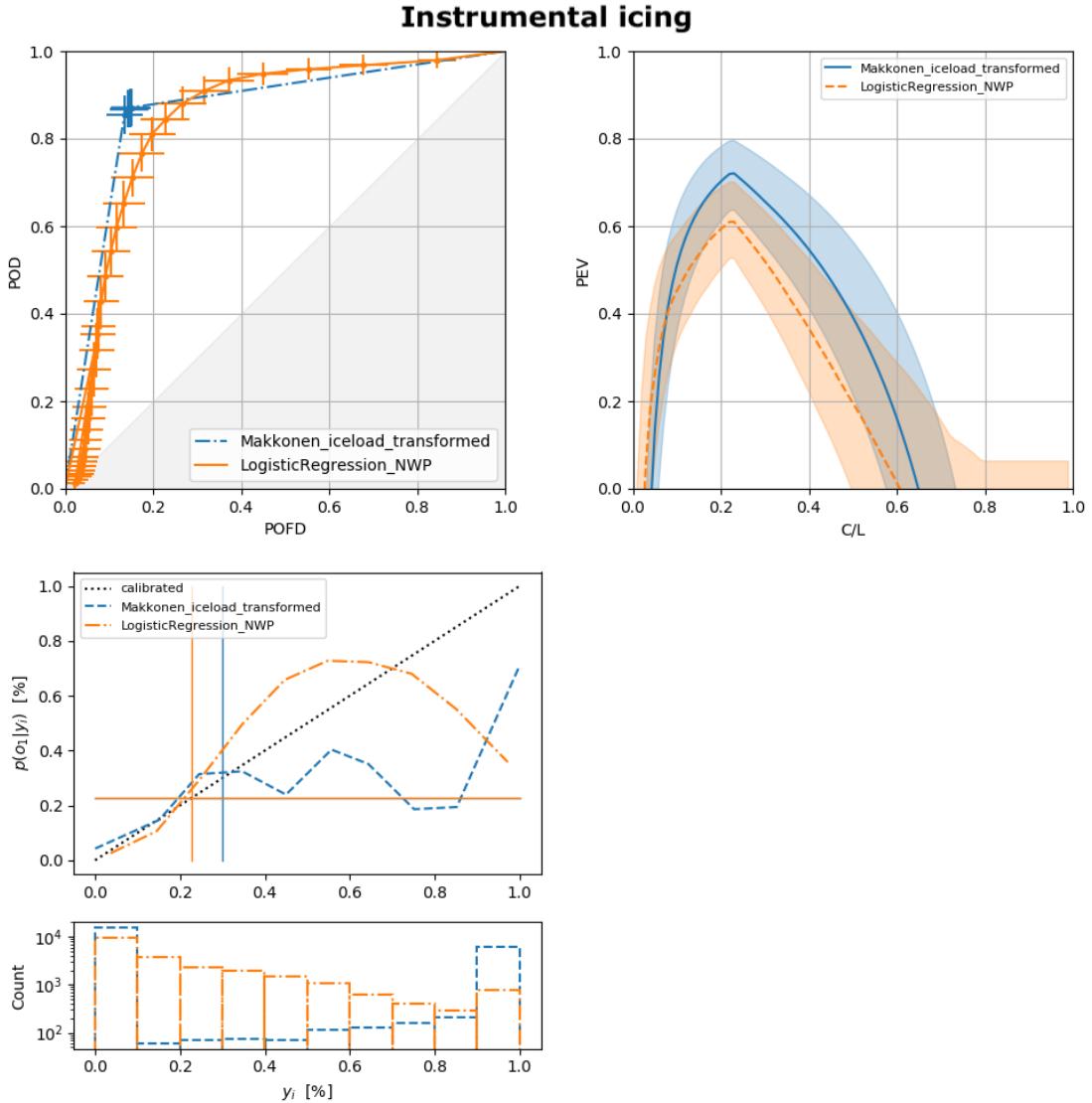


Figure 20: ROC diagram (upper left panel), PEV diagram (upper right panel) and reliability diagram (lower left panel) showing the verification results of the instrumental icing period.

Predicting visible accretion Figure 21 repeats the previous analysis for the visible accretion phase. This time however, the rarity of the event increases the uncertainties. Still, one can see that the transformed Makkonen ice load (blue line) detects more events while having less false alarms compared to the logistic regression that uses all variables (green dot-dashed line). The latter seems to draw its performance from other variables than ice-load, since the logistic regression that uses only ice load as predictor (orange dashed line) is much worse than the

'handcrafted' transformation formula. The logistic regression was not able to derive a comparably performing model of ice load. It seems that at least for logistic regression an appropriate transformation of the input variables is important.

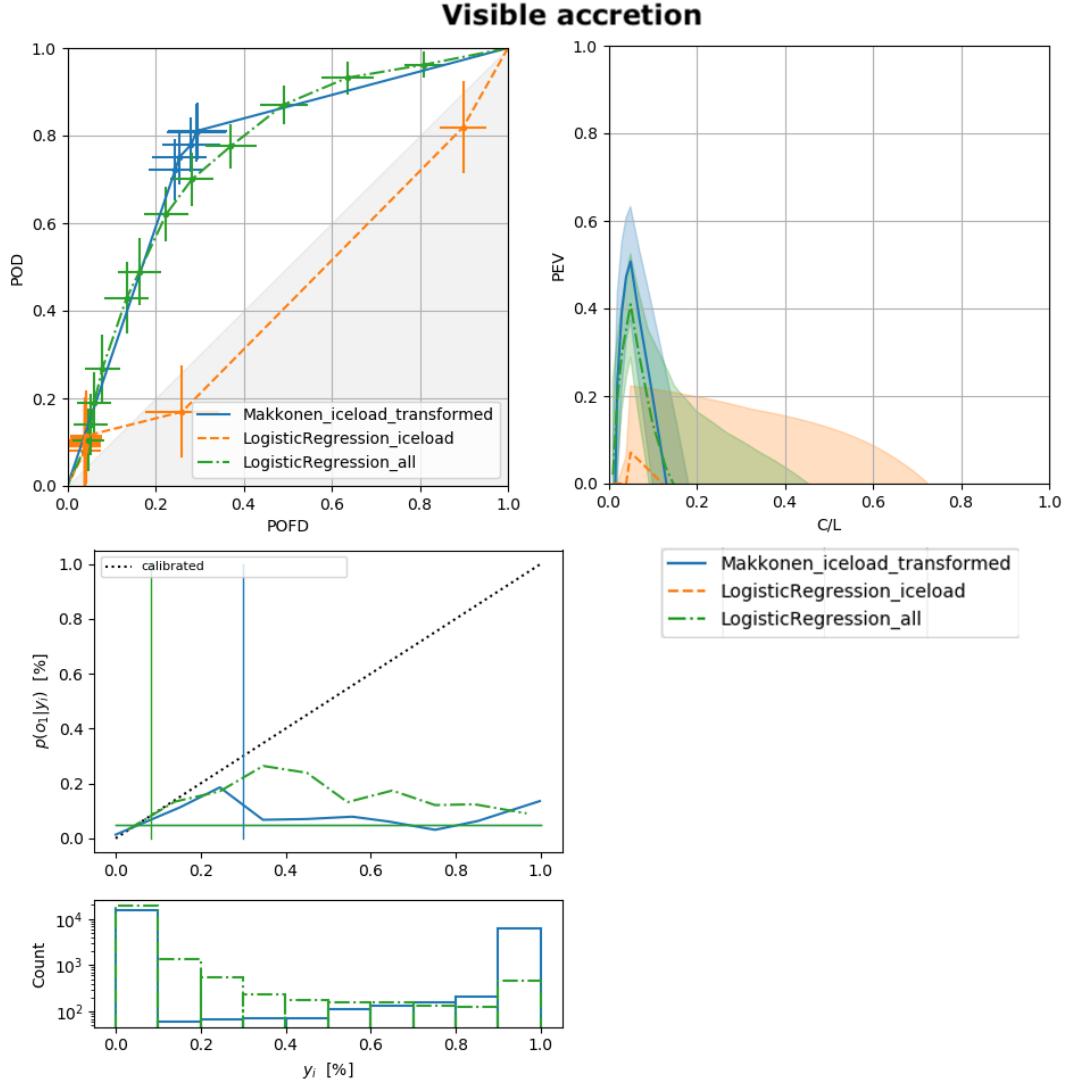


Figure 21: ROC diagram (left upper panel), PEV diagram (right upper panel) and reliability diagram (left lower panel) showing the verification results of the *visible accretion* periods. The worst model is left out in the reliability diagram.

The Makkonen model outperforms the logistic regression in PEV only for $C/L = s$ in a significant way. The PEV for cost-loss ratios higher than 20 % is even negative, which means that forecast users with these cost-loss ratios cannot draw added value from these forecasts.

The reliability diagram shows that both the transformed Makkonen ice load and the logistic regression are not able to reliably predict probabilities above 25%.

This section uses the Makkonen model's ice load parameter. The results using the iceBlade model's ice load were insignificantly different (not shown).

4.3 Adding nonlinearity: Generalized additive models

In this section, the effect of allowing nonlinear functions of the predictors shall be explored by applying the generalized additive model (GAM, see section 3.2.2). This should improve the model's ability to explain a larger fraction of the total variance compared to logistic regression, since classical logistic regression only allows linear dependencies between the predictor variables and the logit of probability.

The improvement of using GAM over logistic regression is given by the difference in verification scores of both models. Additionally, the partial dependency of the logit of probability to each predictor variable is examined.

The predictor variables that were used in this section are temperature, dewpoint spread, wind-speed, cloud-base height, Makkonen and IceBlade icing-rate and ice-load.

Hyperparameter optimization Both models, the logistic regression and the GAM, have one free parameter λ (see equation 3.8) that needs to be determined. This was done by running the cross-validation on monthly blocks. Since the parameter has a different effect in both models, the values may not be compared. For logistic regression, values between 0.1 to 100 were tested, while for GAM, the values were varied between 1 and 1000. Since the validation set was only used to optimize one parameter (λ), the cross-validation scores are expected to be a good estimate of the true out-of-sample performance.

For logistic regression, the parameter does not seem to have a large impact because the best configuration was only about one percent AUC better than the worst configuration, irrespective of the type of icing. For meteorological and instrumental icing, the best AUC was achieved with $\lambda=0.1$ and $\lambda=1$ respectively. For visual accretion, $\lambda=0.1$ produced the highest AUC.

For the GAM, λ had more impact: The best and worst configurations were about 10, 10 and 4 % AUC apart. In meteorological and instrumental icing conditions, $\lambda=5$ and $\lambda=1$ were optimal. For visible accretion, $\lambda=10$ yielded the best AUC. The results of the hyperparameter optimization are also collected in table 4. The GAM was run with 20 base splines.

Verification results The cross-validation scores show that the GAM outperforms logistic regression in meteorological and instrumental icing in both AUC and BSS. For the case of visible accretion, the GAM allows a better discrimination (higher AUC) but does not produce better

	meteorological icing	instrumental icing	visible accretion
Logistic regression	0.1	1	0.1
GAM	5	1	10

Table 4: Hyperparameter optimization: Regularization parameter λ for logistic regression and GAM. Higher values indicate simpler models, i.e. smaller absolute regression coefficients.

calibrated probabilities as the BSS is lower than for logistic regression (using the same BS_{clim}). Even worse, both models' predictions are below zero, which means that their probabilities are worse than climatology.

In terms of improvement, the GAM shows the largest advance over logistic regression (LR) in the case of meteorological icing (fig. 22) in which the GAM detects up to 75 % of the events where the LR detects just 50 % for a POFD of 10 %. This outperformance shows in higher PEV for all cost-loss ratios but significantly higher ones for ratios above 40 %. The GAM also exhibits a better calibration as shown in the reliability diagram (same figure, lower panel) and a better resolution as it gives probabilities near zero or 100 % more often.

For instrumental icing (fig. 23), the GAM also shows improvements over LR for α between 10 and 65 %, but the added value is insignificant compared to the 90 % confidence interval. Here, the calibration curves are quite similar and the GAM showing better resolution.

Both models completely lack value in the visible accretion phase apart from a small neighborhood around $\alpha = s$.

Figure 24 shows that the false alarm rate (the fraction of false alarms of all no-events) is quite high compared to the other curves. Since the visible accretion is about five times more rare than the other phases and comes with double the false alarms at some point, it is preferable to just never take measures and stick with a rare loss L instead of having the cost C for the many false alarms.

The reliability diagram in figure 24 reveals that both models were not able to issue reliable probabilities for visible accretion above 40 %. Interestingly, the GAMs for meteorological and instrumental icing were considerably better at producing calibrated probabilities than the logistic regression.

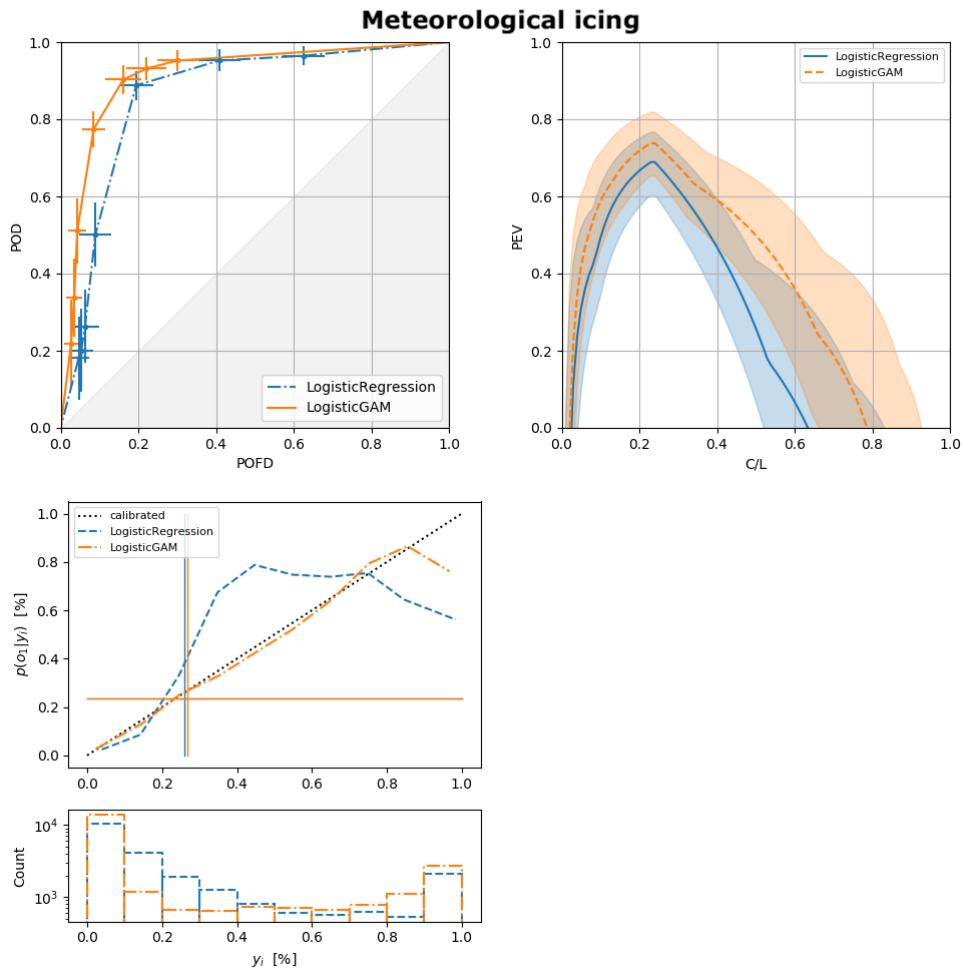


Figure 22: ROC diagram for the GAM and the logistic regression verified against the observed *meteorological icing* periods.

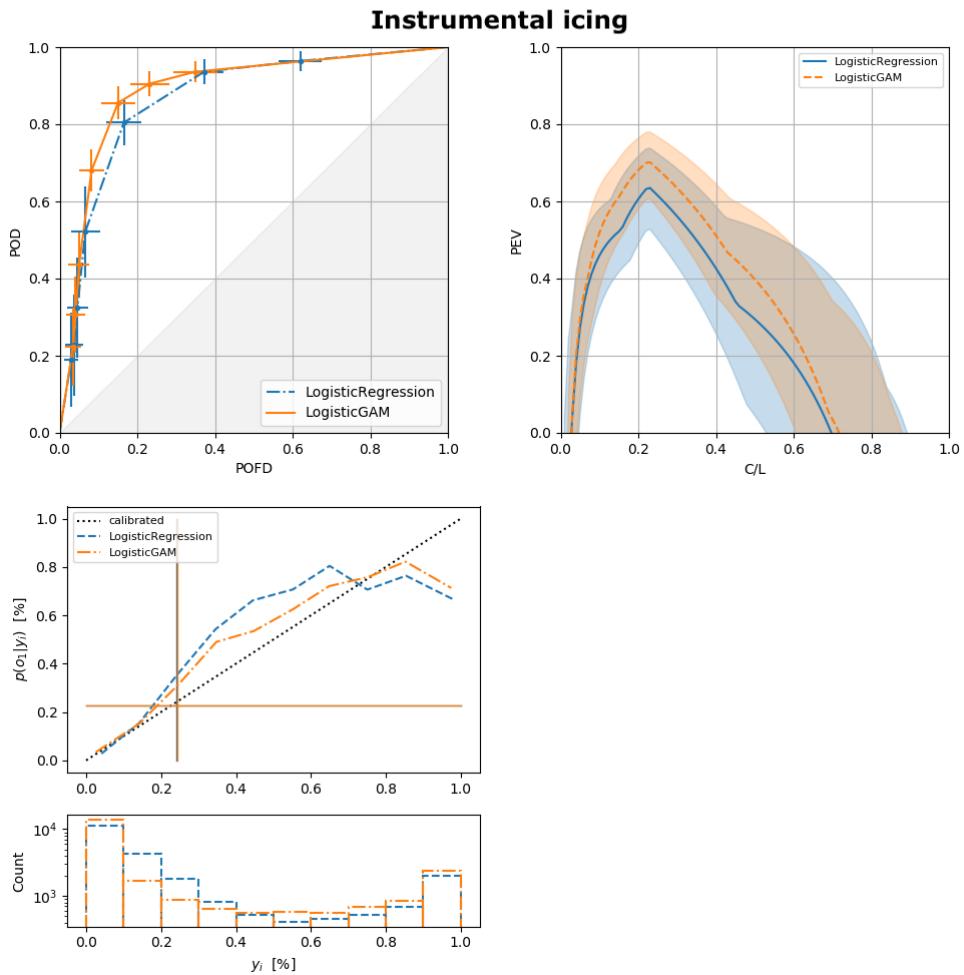


Figure 23: ROC diagram for the GAM and the logistic regression verified against the observed *instrumental icing* periods.

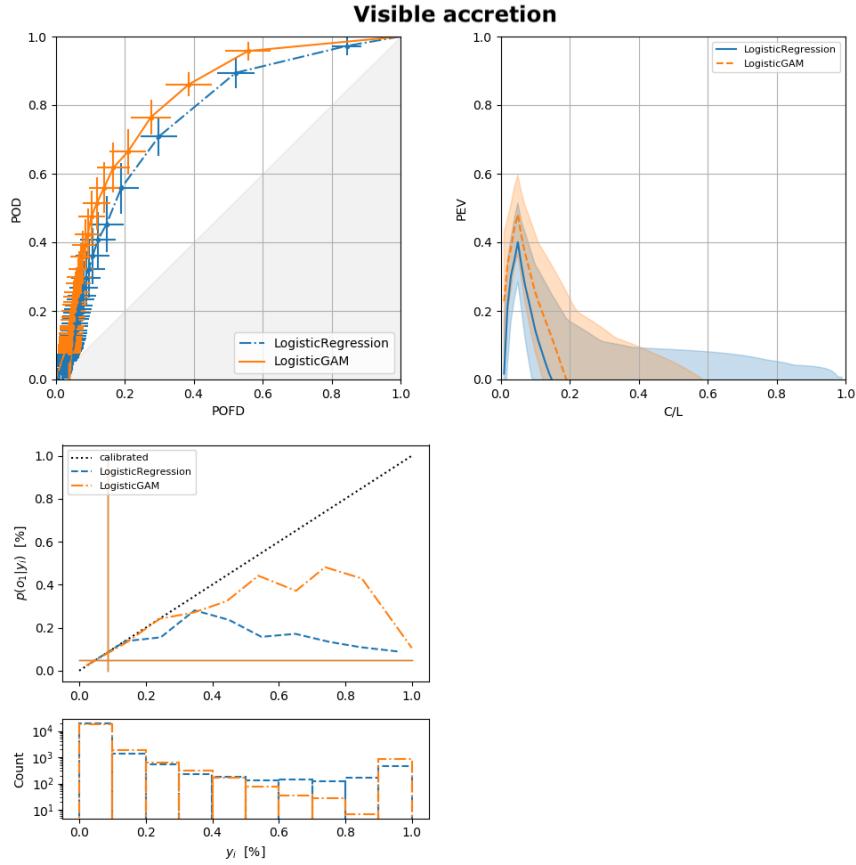


Figure 24: ROC diagram for the GAM and the logistic regression verified against the observed *visible* accretion periods.

Partial dependencies As already pointed out in section 3.2.2, the contribution of each predictor variable in a GAM is independent from all other predictor variables - the partial derivative of the predictand with respect to one predictor variables is independent from all other predictor variables. This characteristic explains the 'additive' in the name of the model and it makes it possible to examine the effect of one variable on the prediction irrespective of all other variables. If we now plot the shape of the additive terms of equation 3.10 into one graph, we have visualized the impact of each variable on the GAM's prediction. This fact makes it a non black-box machine learning model.

Figure 25 shows these partial dependencies, i.e. each predictor's additive contribution in equation 3.10: While temperature is greatly responsible for having a high probability of meteorological and instrumental icing, it is less important in visible accretion, where the spread is much more important than in the other phases. In instrumental icing, the GAM did not find a large contribution from spread but a substantial one from the Makkonen ice-load. This could be

explained by the memory effect of ice-load which removes the mass of ice slowly when icing conditions are not present anymore. A large spread however only describes the atmospheric conditions that do not favor ice accretion but still allow ice to be present at the turbine. Wind speed did not have a large impact in all three cases, just a slight negative effect for higher windspeeds, but some effect may have been captured through the ice-rate variable, which depends on wind too.

To get the impact on the predicted probability, the terms are summed and sigmoid-transformed (equation 3.11). For reference, $\text{sigmoid}(0) = 0.5$, $\text{sigmoid}(1) \approx 0.73$ and $\text{sigmoid}(2) \approx 0.88$. Note that the additive bias term in equation 3.10 has not been plotted.

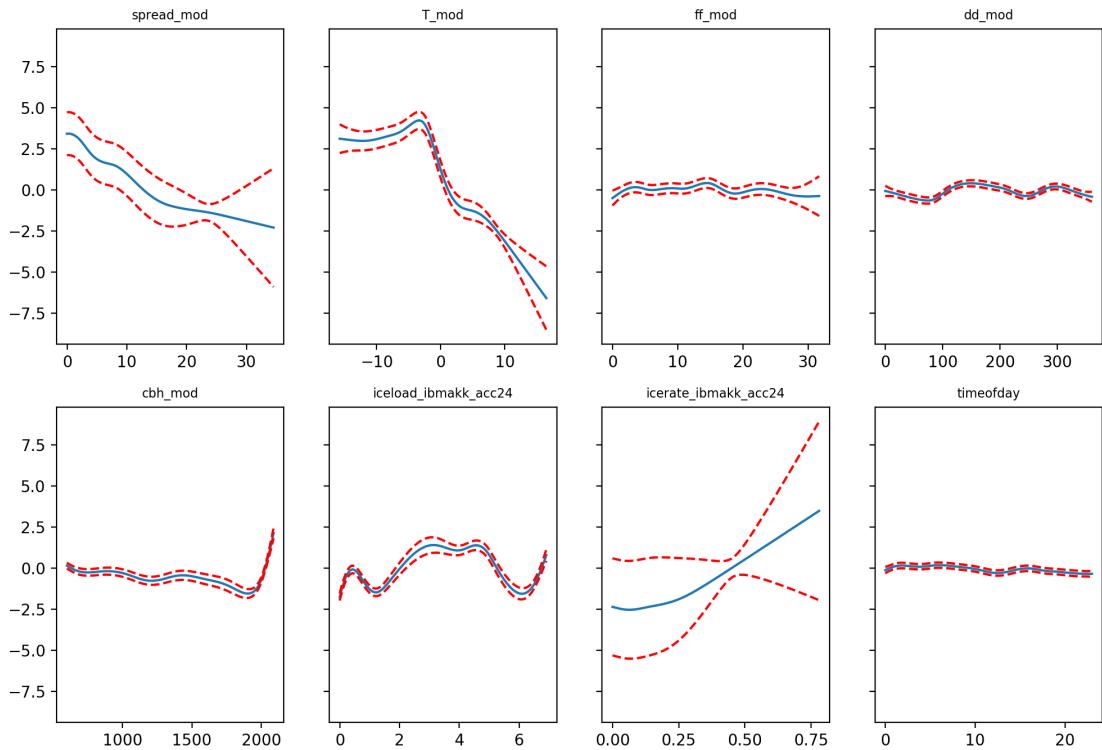


Figure 25: GAM partial dependencies (blue solid lines) for meteorological icing. The red dashed line gives a 90% confidence interval, as computed by the pyGAM package ([Servén and Brummitt, 2018](#)). The variables are, upper row from left to right: Dewpoint spread, temperature, wind speed and direction. Lower row from left to right: cloud-base height, Makkonen ice load and rate, and time-of-day (0-23). The y-axis describes each function's additive component to the logit of probability. The x-axis is in each variable's unit (see table 2).

4.4 Artificial neural nets: An entirely nonlinear approach

Comparing the mathematical structure of the GAM with what we theoretically know about the process of icing, one may notice that icing has two properties which are not realized in the GAM, but both could be implemented with neural nets:

- Firstly, the atmospheric conditions for icing have to be satisfied since the accumulation of ice depends on multiple factors to be present at the same time to a sufficient extent. This motivates a multiplicative structure of the forecast model like the Makkonen model, equation 3.4, such that for example strong wind in combination with low spread causes a high risk of icing while strong wind on its own is not indicative of icing risk when the spread is large. This means that the partial derivative of icing risk with respect to wind speed will depend on spread - something that is not possible for a model that is linear in the predictor variables.
- Secondly, ice-load is not removed instantly when icing conditions are not present anymore. This would suggest that our machine-learning model should have a time dependency too - something that was not realized in this work.

Hyperparameter optimization

As neural nets have many parameters that determine the performance to a great extent, it is greatly beneficial to optimize these parameters. This has been done by cross-validating the model performance of many parameter combinations. Following parameters were varied:

- The main network architecture that is described by the number of layers and nodes per layer. The program representation for the *keras* module uses a tuple with as many entries as there are hidden layers in the network (zero layers means a fully connected network, i.e. a multi-linear regression with a sigmoid activation) with the entry values saying how many nodes are on the respective layer. The output layer, which is the last layer, always has one node - that is one function outputting a scalar value for the predicted probability. The evaluated architectures are: (6,), (32,), (64,), (4,4,4,), (64,32,16,8,)
- A batch size of one and four weeks, as well as using all samples at once (168, 4x168 and 10^5 samples).
- A dropout percentage of 10, 25 and 50 %.
- Kernel regularization, describing how large regression coefficients may get, from 10^{-7} to 10^{-6} .

To address the class imbalance problems, the class distributions were resampled to a ratio of 1:2 and weighting the minority class samples double the majority class ones. The predictor variables were normalized in mean and variance to improve numerical convergence in the optimization procedure, since gradient-based optimization is not scale-invariant. The optimization algorithm *Adam* (Kingma and Ba, 2014) was used since it is widely used in the machine-learning community. The activation function was tanh for hidden layers and sigmoid for the output layer to constrain the output between zero and one.

The calculation of the AUC is performed in *Scikit-learn* v.0.20.3 using the trapezoidal rule.

Meteorological icing The hyperparameter optimization result is visualized in figure 26. The score difference between the tested configurations was lower than expected after fixing an issue in the weight optimization (indicated below). The configuration 'GLMH' (dropout 0.1, kernel_regularizer=1e-7 and architecture (32, 16, 8)) marked the best performance with a validation AUC of 0.969 (0.957-0.978) and a BSS of 0.704 (0.659-0.749). This configuration also happens to have the best BSS, which is not necessarily the case. The crossvalidation scores closely follow the training scores, so overfitting does not seem to be an issue in most cases. However, the initial results showed confusing results which could be tracked down to a wrong setting in the early-stopping part of the weight optimization (fig. 36). This resulted in an overfitting on the validation dataset (validation score higher than training score) and unstable probability predictions (erratically jumping BSS values).

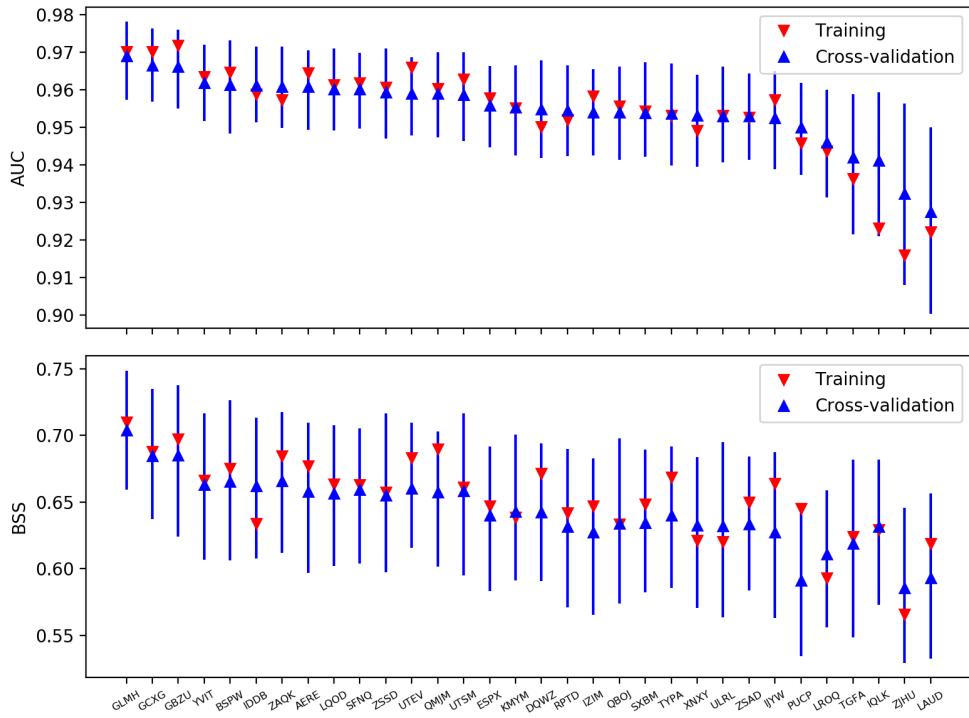


Figure 26: The result of the hyperparameter optimization shows the AUC and BSS for each tested configuration. One tick on the x-axis stands for one parameter configuration marked by a 4-letter identification code, explained in table 5. The AUC is shown in the upper half and the BSS in the lower half of the figure. Red triangles pointing down indicate training scores, blue upward pointing ones mark validation scores. The models are sorted in order of decreasing cross-validation AUC.

Visible accretion The result of the hyperparameter optimization is visualized in figure 27. Apparent is the larger generalization error by the difference between training and cross-validation scores. To improve the prediction of the rare case of visible accretion, the majority class was undersampled from a class distribution of initially 1:20 to 1:2. This reduced the number of training examples considerably from 12177 in the paragraphs above, down to 2772 samples. The experiment 'WXTH' that features the same configuration as in the paragraph above achieved the highest AUC but in this case, for visible accretion, it was associated with a negative BSS. The BSS shows increased variability, presumably due to a small climatological Brier score in the denominator for this rare event.

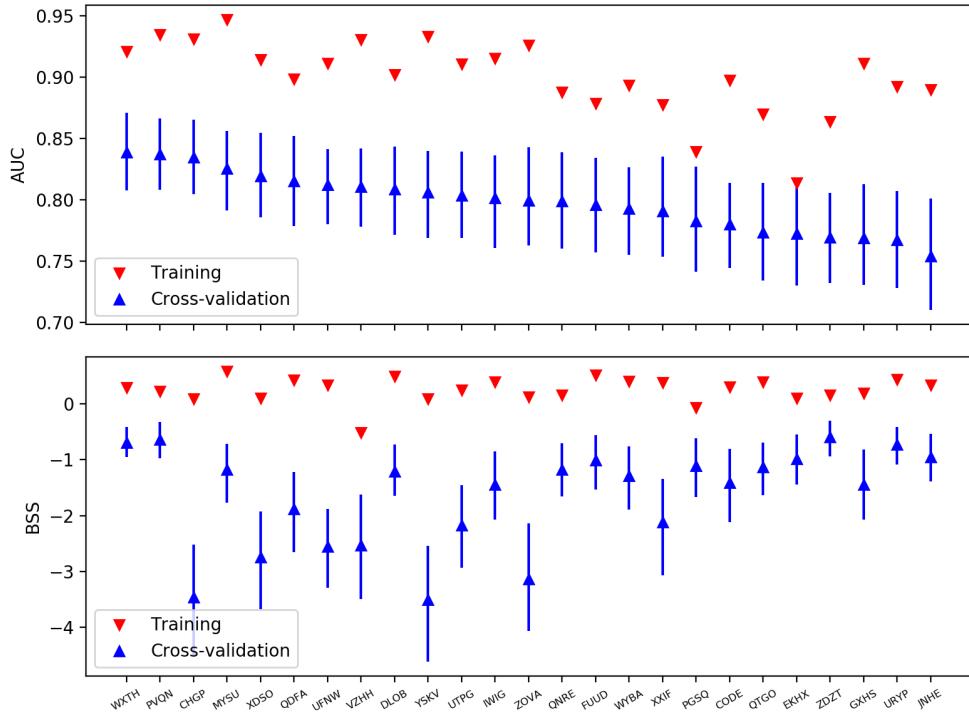


Figure 27: As in figure 26, this time for the visible accretion case. The upper panel shows a significant generalization error which indicates overfitting on the training set. The lower panel shows that the out-of-sample forecasted probabilities are completely unreliable. The configurations are listed in table 6.

Verification results

When using models with a large number of parameters which are estimated from cross-validation, a third 'dataset' is used additionally to training and cross-validation datasets, since the cross-validation scores become biased compared to the out-of-sample scores by choosing the hyperparameters in order to maximize cross-validation scores.

In the present work, the configurations were at first tested on the last 10% of the dataset, but this yielded odd results as the data has a distinct seasonality and so the last 10%, i.e. the last five weeks of the second year of data showed much higher scores than within cross-validation. As explained, the cross-validation results should be biased towards being too optimistic from the theoretical point of view, but this was not the case, so that the results are now presented using cross-validation predictions of the entire dataset of two winter seasons instead of using an extra test set.

Meteorological icing Figure 28 clearly shows the outperformance of the artificial neural net (ANN) compared to the generalized additive model (GAM) over the whole range of cost-loss ratios (upper right panel) through higher detection and lower false alarm rates for all probability thresholds (upper left panel). Even the reliability diagram (lower panel) is clearly in favor of the ANN as it is not significantly over- or underconfident except minor deviations around 45 % predicted probability. The resolution is also much higher for the ANN as mid-probabilities between 40 and 70 % occur only in rare cases (note the logarithmic axis in the histogram). Finally, the error bars are much tighter for the ANN than the GAM.

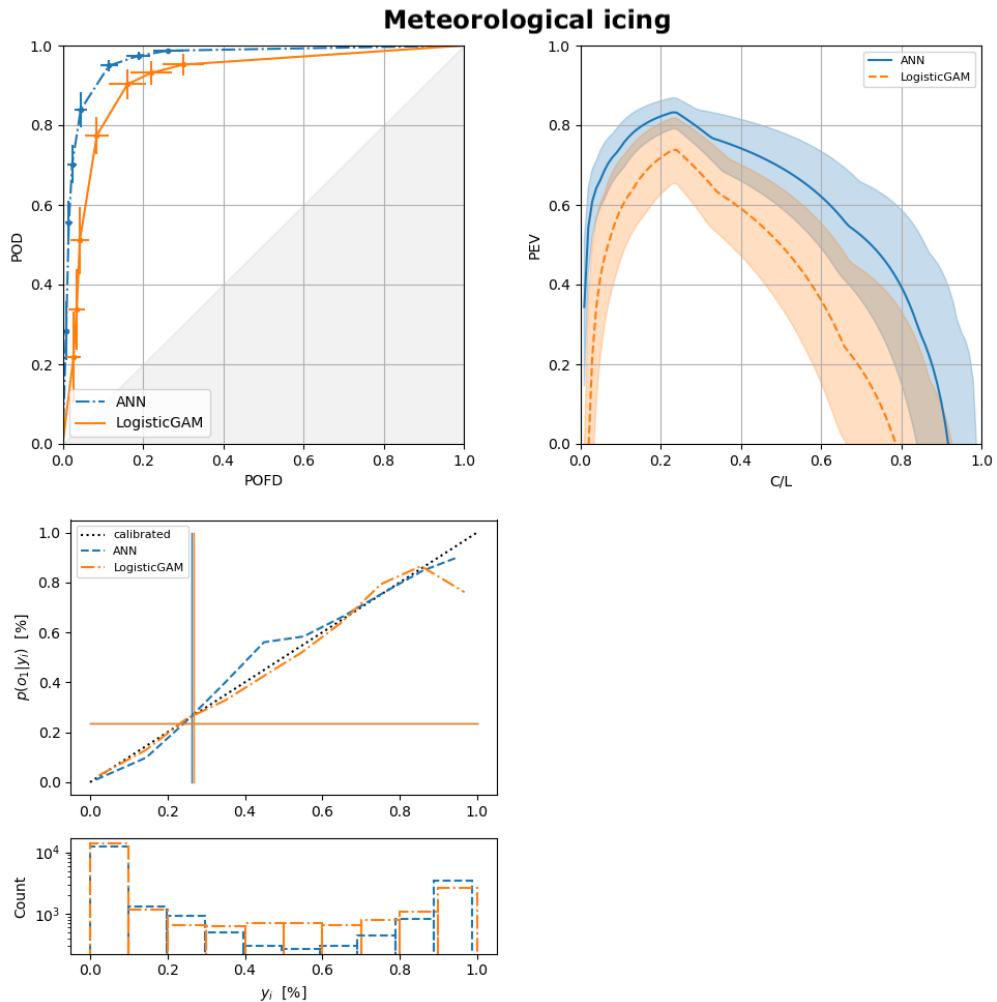


Figure 28: ROC diagram (left upper panel), PEV diagram (right upper panel) and reliability diagram (left lower panel) showing the verification results of the *meteorological icing* phase.

Visible accretion In the visible accretion phase, figure 29, the result is quite the same with the difference that the confidence intervals are much wider due to the rare occurrence of the event. Nevertheless, the outperformance of the ANN is significant but with less confidence. In absolute values of PEV (upper right panel), the predictions of visible accretion are much less potentially valuable compared to the meteorological icing: Values reach 0.6 at $\alpha = s$ but quickly drop to values below 0.2 for $\alpha > 0.2$. For cost-loss ratios above 0.6, the predictions are less valuable than climatology. The reliability diagram in the lower panel shows higher forecasted probabilities in the ANN which results from undersampling the majority class such that the used dataset contains less no-icing cases. While the ANN forecasts reliable probabilities until 70%, the GAM is reliable until 35% only. Interestingly, the ANN never predicts over 90% probability whereas the GAM predicts 90-100% about one thousand times.

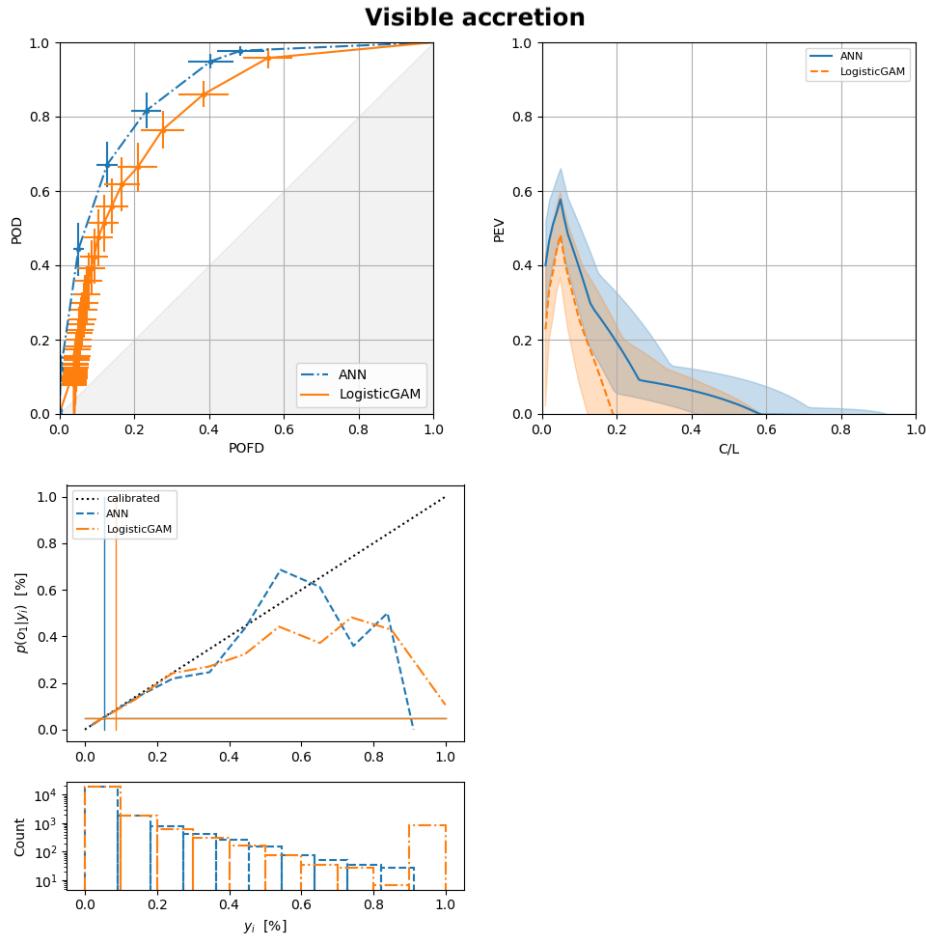


Figure 29: ROC diagram (left upper panel), PEV diagram (right upper panel) and reliability diagram (left lower panel) for the visible accretion periods.

4.4.1 Resampling and regularization to address class imbalance problems

The rarity of the observed visible accretion phase introduces problems in learning the patterns in the predictors leading to icing. Class imbalance may not be the underlying reason for incorrect classifications, as discussed in the introduction section, but easing the imbalance can help. Additionally, regularizing the models should improve the classifications because simpler models focus on the most important patterns, ignoring smaller errors that probably exist due to noise in the data. Both factors are tested here: Resampling and regularization. Resampling randomly selects elements out of the original sample so that a given class ratio is achieved, e.g. 1:10, meaning that there are 10 majority class samples for 1 minority class sample (icing in our case). The decision trees in the next section use a completely balanced approach (1:1).

The effect of both factors is visualized in figure 30. A rare case tends to be detected rarely to minimize the cost function as discussed in the introduction on class imbalance. To increase detection rates, the imbalanced unweighted dataset (orange solid line) is resampled to contain less and less of the majority class examples from initially 1:20 down to 1:10 and to 1:5 sample ratio. Regularization of the weights inside the neural net seems to have comparable effects. In both cases, regularized or not, downsampling the majority class improves the AUC. Additionally, regularization tends to remove false alarms, as indicated by the red line, whereas not regularized version achieves higher detection rates despite higher false alarm rates.

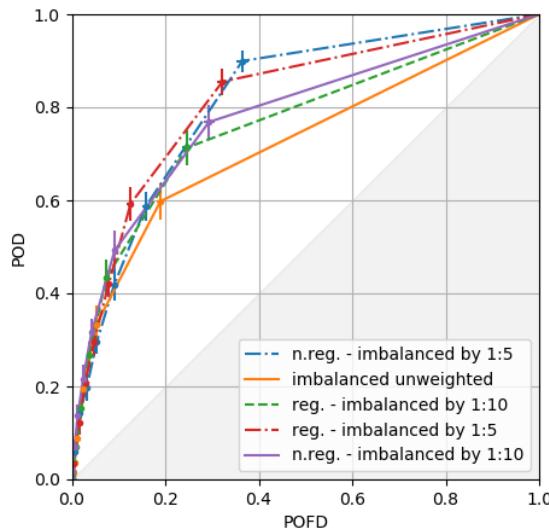


Figure 30: Example of a ROC diagram for resampling and regularization.

Figure 31 shows an example of how the classification could look like in two dimensions. Each point marks a sample in the dataset that is labeled by observations as icing or no-icing.

The location of a point on the two-dimensional plane is given by its temperature and relative humidity. A learning algorithm classifies each point in this space as one of the two classes: Blue regions as icing events, red regions as no-icing. Downsampling the majority class leads to less "red" labeled points in the "correct" regions, but also less red points in the "wrong" region, such that the decision boundary that separates both classes is drawn less close to the blue area, thus more blue labeled points will be detected as blue by the algorithm. The decision boundary in figure 31 is quite smooth, i.e. already regularized. Decreasing regularization would lead to a more wavy line that "fits the noise" in that it curves around every other point.

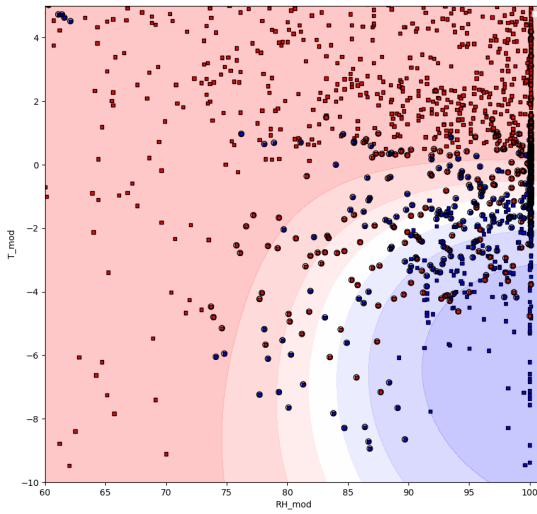


Figure 31: Each point marks a sample in the dataset that is labeled (by observations) as icing or no-icing. The location on the two-dimensional plane is given by its temperature (y-axis) and relative humidity (x-axis). A learning algorithm classifies each point in this space as one of the two classes: Blue regions as icing events, red regions as no-icing.

4.5 Classification trees and support vector machines

Before comparing the results between all the tested models, the best configurations for the remaining models, decision trees and support vector machines, must be found.

Hyperparameter optimization for decision trees The evaluated models cover both major techniques of decision trees: the boosting and the bagging algorithm, implemented in scikit-learn ([Pedregosa et al., 2011](#)) under the name "AdaBoostClassifier" and "BaggingClassifier". The first is an abbreviation for "Adaptive Boosting" ([Freund et al., 1996](#)) and implements an ensemble of decision trees via boosting, i.e. iteratively adding new trees that improve the classification on residual errors. The second one uses "bootstrap aggregation" of the samples to improve

generalization performance (Breiman, 1996). See methods section 3.2.4 for details.

Both algorithms were tested by varying two hyperparameters: 1) the number of trees in the ensemble, where a higher number should cancel out errors by averaging and 2) the algorithm, boosting or bagging. Other parameters were left at their default values: feature bootstrapping for example, that would randomly choose the predictors for each tree, was not used.

The results for meteorological icing show a clear advantage by bagging and confirm the hypothesis of cancelling errors. The best configuration exhibits an AUC of 0.987 (0.982-0.991) and a BSS of 0.816 (0.771-0.855) with 100 trees. Notice that the boosting method is also not designed to producing probabilities, since its output is always zero or one. Consequently, the boosting method's BSS was below zero. The same result was found for visible accretion where the bagging method showed higher AUC and BSS compared to the boosting method. The advantage over boosting is even larger, with differences of up to 9% AUC. In contrast to above, increasing from 50 to 100 estimators did not add any value, as the AUC differed only in the order 10^{-4} . The best model achieved an AUC of 0.857 (0.839-0.878), while the BSS confirmed uncalibrated probabilites at values of -0.41 (-0.66 to -0.17).

Hyperparameter optimization for support vector machines The support vector classification offers two parameters for optimization: the kernel, polynomial or radial-basis-function (RBF), which transforms the input space that is spanned by the predictor variables to another space that is used as input space for the linear SVM algorithm; and the regularization parameter C that leads to a simpler model for larger C.

The results for meteorological icing show an advantage of large regularization for both kernels, the polynomial and the RBF kernel. Additionally, the RBF kernel is clearly more performant: the best configuration with an RBF kernel and C=1000 features an AUC of 0.919 (0.905-0.932) and a BSS of 0.700 (0.644-0.753). The best polynomial kernel SVC achieves only 0.88 AUC and 0.5 BSS.

In the visible accretion case, the RBF has no significant advantage anymore, but still is best with an AUC of 0.678 (0.663-0.695). The BSS is negative again, comparable to the result for the ANN.

Results: Meteorological icing

Now the best three models can be compared: The artificial neural net (ANN), the decision-tree ensemble (with bagging: CART-Bagging) and the support vector classification (SVC).

Surprisingly, the artificial neural net was not the best model for the majority of cost-loss ratios, given the effort to optimize this model. Figure 32 shows that quite the opposite is the case: The

ANN was the worst of the three models for a large range of cost-loss ratios from 0.15 to 0.8. The bagging classifier achieved the highest value for most cost-loss ratios from 0.1 to 0.7 due to its high detection rate and low false alarm rate for all probability thresholds. The SVC reached detection rates that are nearly as high as the CART' ones but due to the SVC's binary prediction, it lacks PEV for cost-loss ratios below 0.2 and above 0.7. The bagging classifier overestimates the probability of icing between 20 and 60 %.

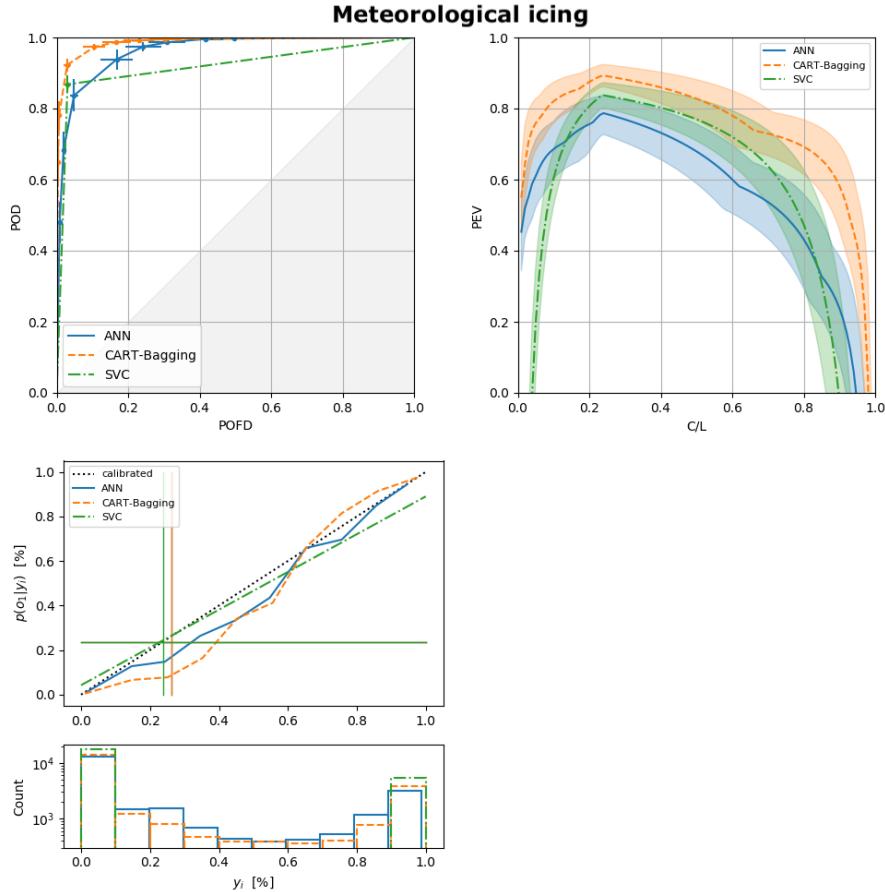


Figure 32: ROC diagram (left upper panel), PEV diagram (right upper panel) and reliability diagram (left lower panel) showing the verification results of the *meteorological icing* phase.

Results: Visible accretion

Contrary to the meteorological icing, the decision-trees were not able to outperform the ANN in any score (figure 33). The ROC diagram displays increased false alarm rates at higher probability thresholds, resulting in lower value for cost-loss ratios above 0.15. The decision-trees'

calibration curve indicates bias to forecast too high probabilities. Compared to the SVC, the decision-trees show less resolution. The SVC's detection rate exceeds the rate of the decision-trees at a certain point but is ultimately worse than the decision trees in PEV since the trees achieve a higher delta in $H - F$ compared to the SVC at lower probability thresholds since the SVC does not provide probability forecasts. The bias of both models is certainly associated with the balanced bagging approach where both classes of outcomes are resampled such that they are equally likely. One could still account for this by rescaling the predicted probability by some factor. Simply dividing by the downsampling factor, i.e. 1:20, would obviously be too much given the calibration curve in figure 33.

To detect 50% of all cases, a false alarm rate of 10% had to be accepted. At a base rate of $s = 0.05$, this means that there were nearly four times as many false alarms as hits, since the hits cover only 2.5% of all cases. This explains the low economic values in the PEV diagram.

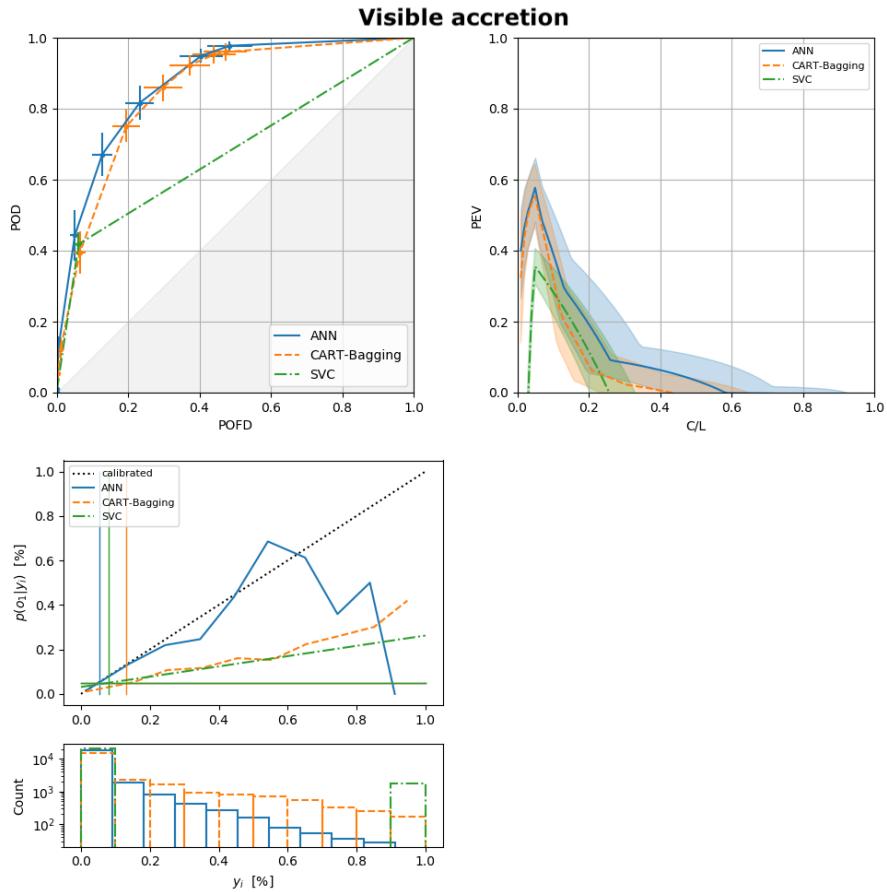


Figure 33: ROC diagram (left upper panel), PEV diagram (right upper panel) and reliability diagram (left lower panel) verifying the *visible* accretion periods.

5 Discussion and Outlook

In the present thesis, a multitude of predictive methods were analyzed by sequentially increasing the complexity of the predictive model and evaluating the effects on forecast quality and value. The results show that the two winter-season dataset was long enough to fit models of increased complexity with a higher predictive accuracy than possible with logistic regression or physical icing models.

Figure 34 summarizes the impact of machine learning on forecasting instrumental icing. The best machine learning model, the decision tree ensemble using bagging, was able to add considerable PEV at a wide range of cost-loss ratios compared to the empirical decision rule and the Makkonen model. Even more, the ML model extended the range of $PEV > 0.5$ from C/L between 10-45 % to 5-95 %. The reason for this increase in PEV lies in a reduction of missed events at low C/L and the reduction of false alarms at high C/L . The visually most prominent feature is the high PEV for C/L around 0.8 which comes from a reduction of the false alarm rate down to 0.5 % while still detecting about 78 % of the events correctly. The reliability diagram (lower left panel) shows that the ML model produced uncalibrated probabilities except for the lowest and highest 10 %. Nevertheless, the ML model's mean forecast probability is close to the observed frequency of the event, indicating low bias.

In contrast, the verification of visible accretion forecasts (figure 35) shows that the machine-learning methods tested in this work add much less PEV compared to the instrumental icing case above. The artificial neural net turned out to predict visible accretion best. Nevertheless, it did not reach $PEV > 0.5$ except around the event's climatological base rate. The reason for the lack of PEV becomes clear after computing the total number of false alarms to misses: For a base rate of 5 %, a POD of 80 % and a POFD of 20 %, there are 19 times more false alarms than misses. According to figure 18, this is optimal only for $C/L \approx 5\%$. For a cost-loss ratio of 0.5, the optimal ratio of false alarms to misses is 1:1, which would require a reduction of POFD to approximately 4 %. Yet, this would be just enough for a PEV slightly greater than zero. To increase the PEV to 0.5, a POFD of less than 2.6 % would be required at a POD of 100 %. This calculation illustrates how difficult the prediction of visible accretion on an hourly basis is.

For meteorological and instrumental icing however, the forecasts are highly valuable to a wide range of cost-loss ratios. In general, the results highlight two important features in the formula for economic value: Firstly, a model has to be able to discriminate the events well, described by the AUC. Secondly, false alarms must be minimized for rare events to achieve high PEV. This can be seen in figure 37, where the PEV curve drops increasingly fast the rarer the event is.

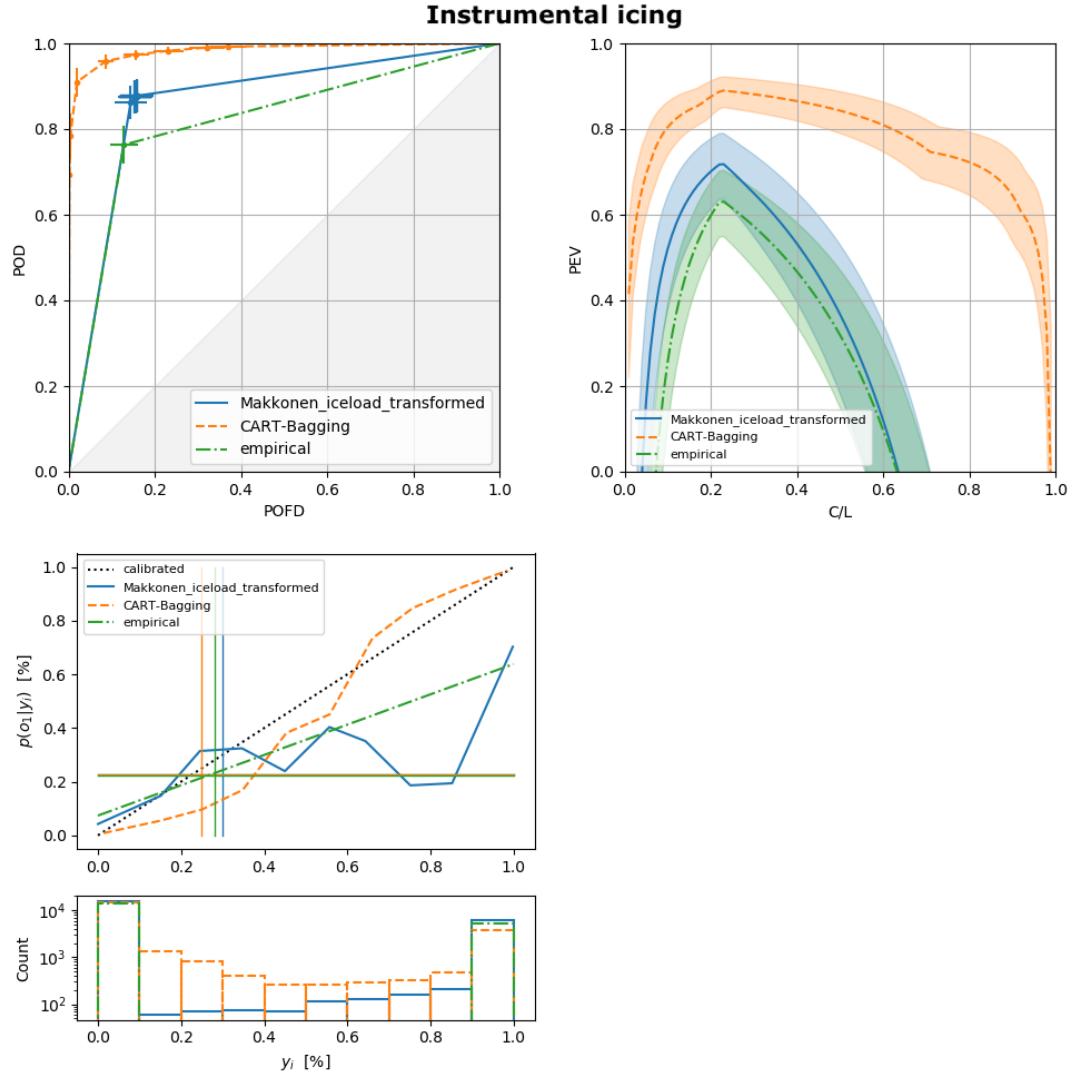


Figure 34: ROC, PEV and reliability diagram showing the potential of machine-learning (Bagging model; orange dashed line) compared to the empirical rule (green dot-dashed line) and the Makkonen model (blue solid line) in forecasting *instrumental icing*.

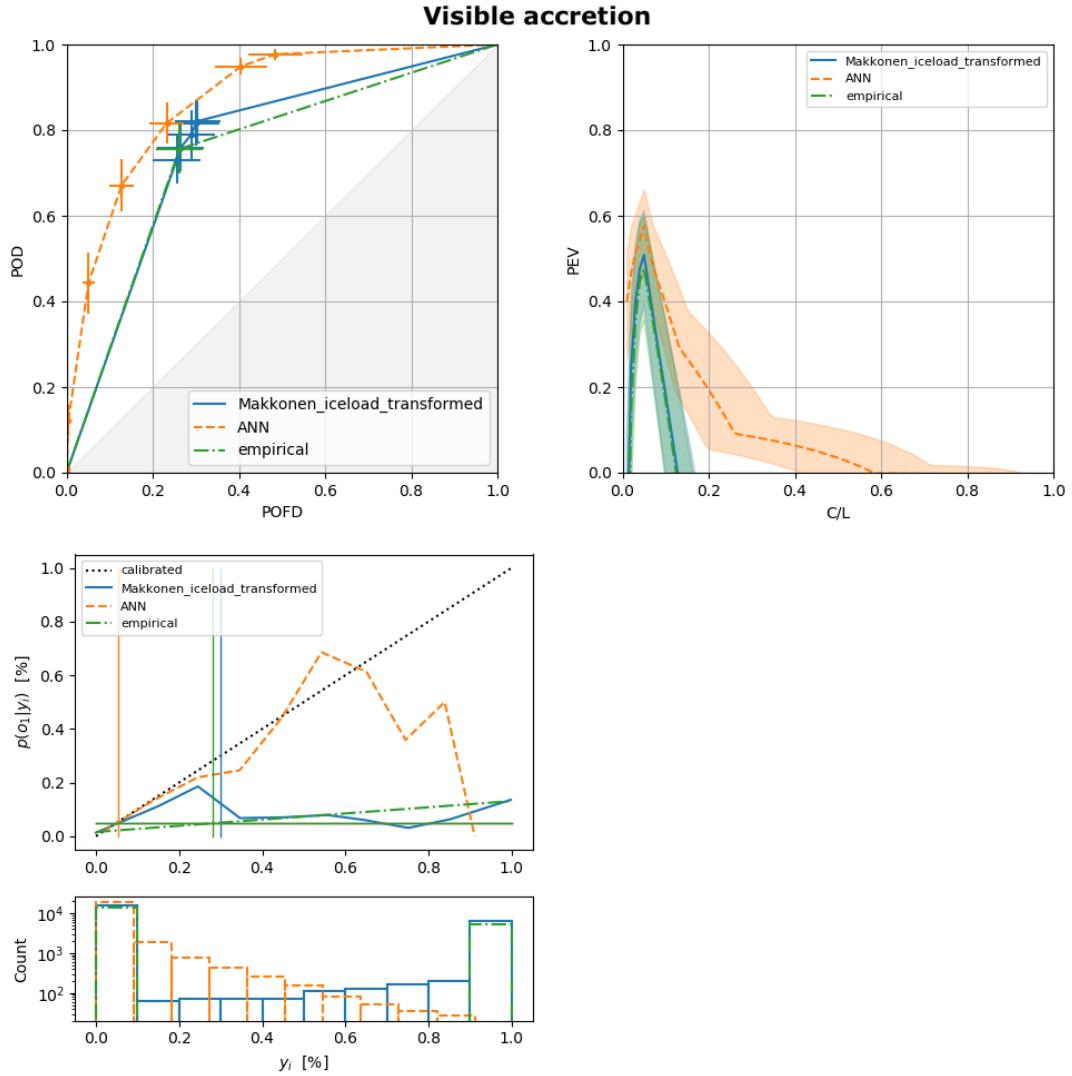


Figure 35: ROC, PEV and reliability diagram showing the potential of machine-learning (ANN; orange dashed line) compared to the empirical rule (green dot-dashed line) and the Makkonen model (blue solid line) in forecasting *visible* accretion.

Machine-learning model interpretation Understanding why a ML model works the way it does is important to establish trust in its predictions. Yet, nonlinearity and non-additiveness in models make it hard to visualize partial dependencies of each predictor variable. Nevertheless there are many techniques to gain insight into the structure of black-box models, for example by approximating a black-box model with a non-black box model (i.e. a plain decision tree). The non-black box model can then be used to explain the behavior of black-box models to some extent. Still, what makes the black-box models more powerful than explainable models is their

predictive accuracy. If they would not give significantly better predictions, why should one use them in the first place? [Vovk et al. \(2015\)](#), ch. 19, discusses why it is so hard to interpret predictive models like the SVM.

Outlook Given the relatively poor results in the case of visible accretion in sections 4.5 one can argue that the model is not reliably able to predict this type of event given the NWP data, the in-situ observations at hand, the choice of predictor variables and the models used. Since the NWP data and the in-situ observations are generally assumed to be reliable, the focus could be on the last two factors: If the current prediction error is mostly associated with synoptic and cloud-scale uncertainty in the atmospheric parameters, then it would be beneficial to use the ensemble data by directly using ensemble mean and standard-deviation as input variables ([Rasp and Lerch, 2018](#)) or adapting the model to use all ensemble members instead of just one. If the prediction error is not due to errors in the atmospheric conditions, the model may be unable to capture the events by design.

Recurrent neural networks should be tested because turbine icing is an inherently time-dependent process and these network architectures are specifically designed to utilize time dependencies in the data.

Bayesian optimization ([Snoek et al., 2012](#)) could be used in the future to obtain better and more robust results since an exhaustive search (i.e. testing all configurations) is not practical when a larger number of hyperparameters need to be estimated, e.g. when using neural nets.

Besides evaluating a complex black-box model on a possibly too short test dataset, one could attempt another way by creating synthetic but realistic datasets for all physically imaginable cases of icing or apply the model to reanalysis output of similar climatology to get a more comprehensive picture of the performance. Of course, the prediction timeseries could not be evaluated by comparison to observations as these are not available, but by checking the plausibility of icing occurrence statistically as a function of other atmospheric variables or by anomaly detection methods.

References

- Abadi, M., and Coauthors, 2016: Tensorflow: A system for large-scale machine learning. *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, 265–283, URL <https://www.usenix.org/system/files/conference/osdi16/osdi16-abadi.pdf>.
- Abu-Mostafa, Y. S., M. Magdon-Ismail, and H.-T. Lin, 2012: *Learning from data*, Vol. 4. AMLBook New York, NY, USA:.
- Andersen, E., E. Börjesson, P. Vainionpää, and L. S. Undem, 2011: Wind power in cold climate. *WSP Environmental*.
- Benjamin, S. G., and Coauthors, 2016: A north american hourly assimilation and model forecast cycle: The rapid refresh. *Monthly Weather Review*, **144** (4), 1669–1694.
- Breiman, L., 1996: Bagging predictors. *Machine learning*, **24** (2), 123–140.
- Breiman, L., and Coauthors, 2001: Statistical modeling: The two cultures (with comments and a rejoinder by the author). *Statistical science*, **16** (3), 199–231.
- Chollet, F., and Coauthors, 2015: Keras. <https://keras.io>.
- Davis, N., A. N. Hahmann, N.-E. Clausen, and M. Žagar, 2014a: Forecast of icing events at a wind farm in sweden. *Journal of Applied Meteorology and Climatology*, **53** (2), 262–281.
- Davis, N., A. N. Hahmann, N.-E. Clausen, and M. Zagar, 2014b: Icing impacts on wind energy production. Ph.D. thesis.
- Davis, N. N., P. Pinson, A. N. Hahmann, N.-E. Clausen, and M. Žagar, 2016: Identifying and characterizing the impact of turbine icing on wind farm power generation. *Wind Energy*, **19** (8), 1503–1518.
- Drage, M. A., and G. Hauge, 2008: Atmospheric icing in a coastal mountainous terrain. measurements and numerical simulations, a case study. *Cold Regions Science and Technology*, **53** (2), 150–161.
- Freund, Y., R. E. Schapire, and Coauthors, 1996: Experiments with a new boosting algorithm. *icml*, Citeseer, Vol. 96, 148–156.
- Frohboese, P., and A. Anders, 2007: Effects of icing on wind turbine fatigue loads. *Journal of Physics: Conference Series*, IOP Publishing, Vol. 75, 012061.
- Géron, A., 2017: *Hands-on machine learning with Scikit-Learn and TensorFlow: concepts, tools, and techniques to build intelligent systems*. " O'Reilly Media, Inc.".
- Haixiang, G., L. Yijing, J. Shang, G. Mingyun, H. Yuanyue, and G. Bing, 2017: Learning from class-imbalanced data: Review of methods and applications. *Expert Systems with Applications*, **73**, 220–239.
- Hastie, T., R. Tibshirani, and J. Friedman, 2009: *The Elements of Statistical Learning*, Vol. 2. Springer, 658 pp.
- Hastie, T., R. Tibshirani, and R. J. Tibshirani, 2017: Extended comparisons of best subset selection, forward stepwise selection, and the lasso. *arXiv preprint arXiv:1707.08692*.
- Hastie, T. J., and R. J. Tibshirani, 1990: Generalized additive models. *Monographs on Statistics and Applied Probability*, **43**.
- He, H., and Y. Ma, Eds., 2013: *Imbalanced learning: Foundations, Algorithms, and Applications*. 1st ed., John Wiley & Sons.

- Holte, R. C., L. Acker, B. W. Porter, and Coauthors, 1989: Concept learning and the problem of small disjuncts. *IJCAI*, Citeseer, Vol. 89, 813–818.
- Hsieh, W., 2009: *Machine learning methods in the environmental sciences: Neural networks and kernels*. Cambridge university press.
- Ioffe, S., and C. Szegedy, 2015: Batch normalization: Accelerating deep network training by reducing internal covariate shift. *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, JMLR.org, 448–456, ICML’15, URL <http://dl.acm.org/citation.cfm?id=3045118.3045167>.
- Jasinski, W. J., S. C. Noe, M. S. Selig, and M. B. Bragg, 1998: Wind turbine performance under icing conditions. *Journal of Solar Energy Engineering*, **120** (1), 60–65.
- Jolliffe, I. T., and D. B. Stephenson, 2012: *Forecast verification: a practitioner’s guide in atmospheric science*. John Wiley & Sons.
- Kingma, D. P., and J. Ba, 2014: Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- Lehtomäki, V., 2016: Wind energy in cold climates available technologies - report. Tech. rep., IEA Wind.
- López, V., A. Fernández, S. García, V. Palade, and F. Herrera, 2013: An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. *Information sciences*, **250**, 113–141.
- Makkonen, L., 1984: Modeling of ice accretion on wires. *Journal of climate and applied meteorology*, **23** (6), 929–939.
- Makkonen, L., 2000: Models for the growth of rime, glaze, icicles and wet snow on structures. *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, **358 (1776)**, 2913–2939.
- Pedregosa, F., and Coauthors, 2011: Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, **12**, 2825–2830.
- Provost, F., 2000: Machine learning from imbalanced data sets 101. *Proceedings of the AAAI’2000 workshop on imbalanced data sets*, AAAI Press, Vol. 68, 1–3.
- Racine, J., 2000: Consistent cross-validatory model-selection for dependent data: hv-block cross-validation. *Journal of econometrics*, **99** (1), 39–61.
- Rasp, S., and S. Lerch, 2018: Neural networks for postprocessing ensemble weather forecasts. *Monthly Weather Review*, **146**, doi:10.1175/MWR-D-18-0187.1.
- Rasp, S., M. S. Pritchard, and P. Gentine, 2018: Deep learning to represent subgrid processes in climate models. *Proceedings of the National Academy of Sciences*, **115**, 201810286, doi:10.1073/pnas.1810286115.
- Reinsch, C. H., 1967: Smoothing by spline functions. *Numerische mathematik*, **10** (3), 177–183.
- Richardson, D. S., 2000: Skill and relative economic value of the ECMWF ensemble prediction system. *Quarterly Journal of the Royal Meteorological Society*, **126 (563)**, 649–667.
- Ronsten, G., and Coauthors, 2012: State-of-the-art of wind energy in cold climates. *IEA Wind Task XIX, VTT, Finland*.

- Scher, S., and J. Molinder, 2019: Machine learning-based prediction of icing-related wind power production loss. *IEEE Access*, **7**, 129 421–129 429.
- Schneider, M., 2017: Case studies of wind turbine icing at a wind farm in ellern, germany, during winter 2016/17, BSc. thesis.
- Servén, D., and C. Brummitt, 2018: pygam: Generalized additive models in python v0.8.0. URL <http://doi.org/10.5281/zenodo.1476122>.
- Snoek, J., H. Larochelle, and R. P. Adams, 2012: Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 2951–2959.
- Srivastava, N., G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, 2014: Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, **15** (1), 1929–1958.
- Stanski, H., L. Wilson, and W. Burrows, 1989: Survey of common verification methods in meteorology. Tech. Rep. 8, World Meteorological Organization.
- Strauss, L., S. Serafin, and M. Dorninger, 2019: Skill and potential economic value of forecasts of active icing for wind turbine operation, to be submitted.
- Thompson, G., B. E. Nygaard, L. Makkonen, and S. Dierer, 2009: Using the Weather Research and Forecasting (WRF) model to predict ground/structural icing. *13th International Workshop on Atmospheric Icing on Structures, METEOTEST, Andermatt, Switzerland*.
- Tikhonov, A. N., 1963: On the solution of ill-posed problems and the method of regularization (russian). *Doklady Akademii Nauk, Russian Academy of Sciences*, Vol. 151, 501–504.
- Vapnik, V. N., 1999: An overview of statistical learning theory. *IEEE transactions on neural networks*, **10** (5), 988–999.
- Vapnik, V. N., and A. Y. Chervonenkis, 2015: On the uniform convergence of relative frequencies of events to their probabilities. *Measures of complexity*, Springer, 11–30.
- Virk, M. S., M. C. Homola, and P. J. Nicklasson, 2010: Effect of rime ice accretion on aerodynamic characteristics of wind turbine blade profiles. *Wind Engineering*, **34** (2), 207–218.
- Vovk, V., H. Papadopoulos, and A. Gammerman, 2015: *Measures of Complexity: Festschrift for Alexey Chervonenkis*. Springer.
- Weiß, C., 2018: Quantifying the uncertainty of empirical models for the icing on structures and wind turbine blades. M.S. thesis, University of Vienna.
- Weissinger, M., 2017: Synoptische Analyse von Vereisungsfällen an Windkraftanlagen am Beispiel Ellern, Deutschland, BSc. thesis.
- Wilks, D., 1997: Resampling hypothesis tests for autocorrelated fields. *Journal of Climate*, **10** (1), 65–82.
- Wilks, D., 2011: *Statistical Methods in the Atmospheric Sciences*, International Geophysics, Vol. 100. 3rd ed., Academic Press.

List of Figures

1	Definition of the phases of icing	7
2	Geographical domain of the NWP model	9
3	An overview of post-processing and learning from observations	10
4	Tradeoff between under- and overfit	13
5	Bias-Variance example	15
6	Out-of-sample error for increasing model complexity	16
7	Comparison of concepts: Generalization error and bias-variance tradeoff	16
8	Bias-Variance effect on simple and complex models	17
9	Cross-validation of correlated timeseries	18
10	GAM: Example of spline basis functions	22
11	ANN architecture: The fully connected network	23
12	ANN architecture: The multi-layer perceptron	24
13	SVC algorithm construction	27
14	SVC: Maximum margin classifier	27
15	SVC: Non-separable case	28
16	PEV diagram construction	32
17	PEV: The expense matrix	33
18	PEV diagram: Effect of varying the probability threshold	35
19	Empirical icing classification: ROC & PEV diagram	38
20	Verification: Makkonen versus Logistic regression on NWP	40
21	Verification of the Makkonen model predictions	41
22	GAM: Verification of the meteorological icing phase	44
23	GAM: Verification of the instrumental icing phase	45
24	GAM: Verification of the visible accretion phase	46
25	GAM: partial dependencies for meteorological icing	47
26	ANN: Hyperparameter optimization meteorological icing	50
27	ANN: Hyperparameter optimization visible accretion	51
28	ANN: Verification of the meteorological icing phase	52
29	ANN: Verification of the visible accretion phase	53
30	Example of a ROC diagram for resampling and cass weighting	54
31	Example of a two-dimensional decision boundary	55
32	Comparison of CART, SVC, meteorological icing phase	57
33	ANN: Verification of the visible accretion phase	58
34	The added value of ML in forecasting instrumental icing	60
35	The added value of ML in forecasting visible accretion	61
36	ANN hyperparameter optimization overfitted on validation set	70
37	Effect of base rate on PEV	70

List of Tables

1	Observational data: variables and their respective measurement instruments.	7
2	The used input variables including derived ones and their physical units	8
3	Verification results for the empirical icing classification	38
4	Logistic regression and GAM: Hyperparameter optimization	43
5	Hyperparameter optimization: ANN configurations for meteorological icing . . .	71
6	Hyperparameter optimization: ANN configurations for visible accretion	72

A Appendix

Acknowledgements

The idea for this thesis originated from Ass.-Prof. Manfred Dorninger and Dr. Lukas Strauss. A team at the department of Meteorology and Geophysics, including Dr. Stefano Serafin, took part in the ICE CONTROL project (2016-2019) which was led by the Central Institute for Meteorology und Geodynamics (ZAMG) and funded by the Austrian Research Promotion Agency (Österreichische Forschungsförderungsgesellschaft mbH, abbreviated FFG). This project provided the opportunity to work with actual wind-turbine data, which is absolutely pivotal for a post-processing topic.

Big thanks go to Ass.-Prof. Dr. Manfred Dorninger and Dr. Lukas Strauss for supervising the thesis, as well as for their feedback and questions to help me to stay on course throughout the time.

Furthermore I would like to acknowledge Dr. Strauss for preparing the forecast and observation datasets into NetCDF files and Dr. Strauss and Dr. Serafin for providing their huge code base for verification, moreover Sebastian Lehner for inspiring discussions and proof-reading.

Lastly I thank my family, my mother and my father, for all their support.

Software acknowledgements

Since it is unfeasible to implement the ML models from scratch, these packages were used: *Scikit-learn* v.0.20.3 (Pedregosa et al., 2011), *pyGAM* v.0.7.1 (Servén and Brummitt, 2018) and *keras* (Chollet et al., 2015) with *Tensorflow* v.1.1.0 (Abadi et al., 2016) as backend.

Change in economic value by hits and false alarms

The economic value in terms of hits a , misses c , false alarms b , correct rejections d , base rate s and total forecasts n for $\alpha > s$ is

$$V = \frac{\frac{a+c}{n}L - \frac{a+b}{n}C - \frac{c}{n}L}{s(L-C)} \quad \text{if } \alpha > s \quad (\text{A.1})$$

Then, the change in V by moving one correct rejection to a false alarm ($d \rightarrow b$) is

$$\Delta V|_{d \rightarrow b} = \frac{-C/n}{s(L-C)} = \frac{1}{ns} \frac{\alpha}{1-\alpha} \quad (\text{A.2})$$

and the change in V by moving one miss to one hit ($c \rightarrow a$) is

$$\Delta V|_{c \rightarrow a} = \frac{-C/L + cL/n}{s(L-C)} = \frac{1}{ns} \quad (\text{A.3})$$

For $\alpha < s$, the economic value changes due to the climatological expense E_{clim} to

$$V = \frac{C - \frac{a+b}{n}C - \frac{c}{n}L}{C(1-s)} \quad \text{if } \alpha < s \quad (\text{A.4})$$

so that the change in V swaps between both effects:

$$\Delta V|_{d \rightarrow b} = \frac{L-C}{nC(1-s)} = \frac{1}{n(1-s)} \frac{1-\alpha}{\alpha} \quad (\text{A.5})$$

$$\Delta V|_{c \rightarrow a} = \frac{-C/n}{C(1-s)} = -\frac{1}{n(1-s)} \quad (\text{A.6})$$

The ratio of both effects is $(1-\alpha)/\alpha$ for $\alpha < 0.5$ and $\alpha/(1-\alpha)$ else.

The derivative of economic value with respect to hit rate and false alarm rate is:

$$\frac{\partial V}{\partial H} = \frac{(1-\alpha)s}{\min(s, \alpha) - s\alpha} \quad (\text{A.7})$$

$$-\frac{\partial V}{\partial F} = \frac{(1-s)\alpha}{\min(s, \alpha) - s\alpha} \quad (\text{A.8})$$

Both expressions balance when $\alpha = s$, which means that one percent increase in hit rate has the same effect as one percent decrease in false alarm rate.

Additional resources

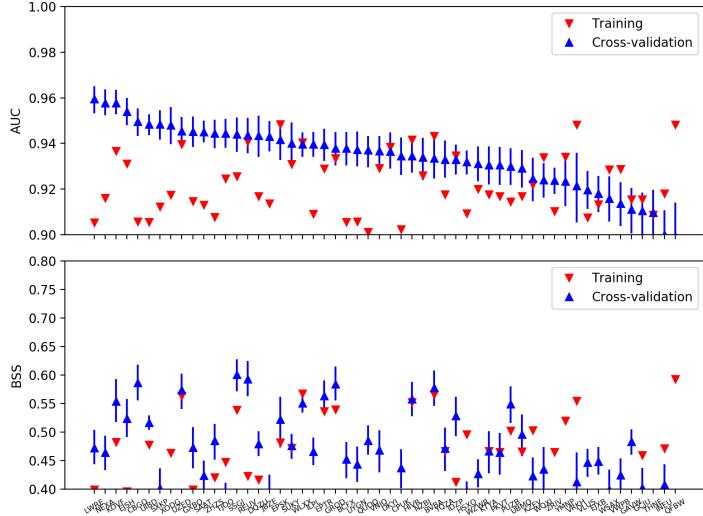


Figure 36: ANN hyperparameter optimization with the *keras* early-stopping option activated (`restore_best_weights=True`). With this option, the weights are taken which were best on the validation set during training, leading to higher AUC but also irregular and unstable BSS values. This image was replaced by figure 26 due to its confusing result.

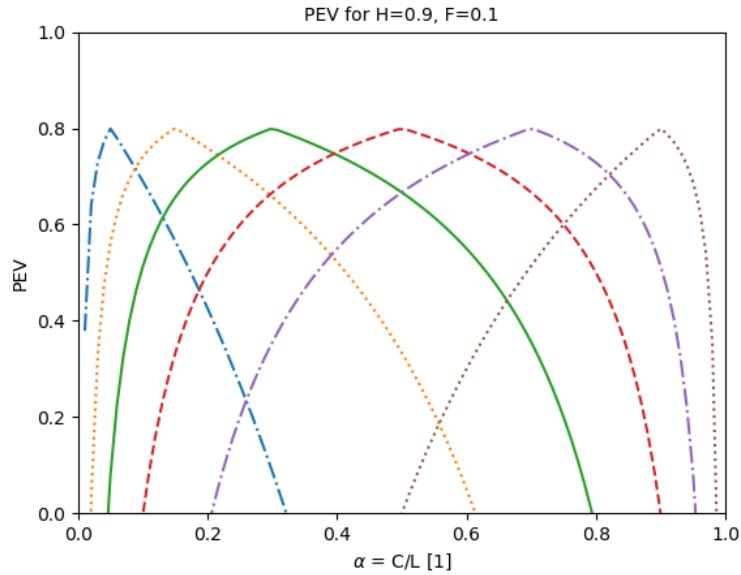


Figure 37: The effect of the climatological base rate on the shape of the PEV curve at a hit rate of 90% and a false alarm rate of 10% given different base rates.

modelcode	N_hidden_nodes	dropout	kernel_regularizer
GLMH	[32, 16, 8]	0.1	1e-07
GCXG	[32, 16, 8]	0.1	1e-06
GBZU	[32, 16, 8]	0.1	1e-06
YVIT	[32]	0.1	1e-07
BSPW	[64]	0.1	1e-06
IDDB	[32]	0.1	1e-06
ZAQK	[32, 16, 8]	0.25	1e-07
AERE	[64]	0.1	1e-07
LQOD	[64]	0.25	1e-06
SFNQ	[32]	0.25	1e-06
ZSSD	[64]	0.25	1e-07
UTEV	[32, 16, 8]	0.25	1e-06
QMJM	[32]	0.25	1e-07
UTSM	[32, 16, 8]	0.25	1e-06
ESPX	[64]	0.5	1e-06
KMYM	[6]	0.1	1e-06
DQWZ	[4, 4, 4]	0.1	1e-06
RPTD	[6]	0.25	1e-06
IZIM	[64]	0.5	1e-07
QBOJ	[32]	0.5	1e-06
SXBM	[6]	0.1	1e-07
TYPA	[4, 4, 4]	0.1	1e-06
XNXY	[6]	0.25	1e-06
ULRL	[6]	0.1	1e-06
ZSAD	[6]	0.25	1e-07
IJJYW	[4, 4, 4]	0.1	1e-07
PUCP	[32, 16, 8]	0.5	1e-06
LROQ	[6]	0.5	1e-06
TGFA	[4, 4, 4]	0.25	1e-06
IQLK	[4, 4, 4]	0.25	1e-06
ZJHU	[4, 4, 4]	0.25	1e-07
LAUD	[4, 4, 4]	0.5	1e-06

Table 5: All configurations tested in the hyperparameter optimization for the ANN for meteorological icing. The 'modelcode' is an internal code identifying a specific configuration. 'kernel_regularizer' is the amount of L2 regularization of ANN weights.

modelcode	N_hidden_nodes	dropout	kernel_regularizer
WXTH	[32, 16, 8]	0.1	1e-07
PVQN	[32, 16, 8]	0.1	1e-06
CHGP	[64]	0.1	1e-06
MYSU	[32, 16, 8]	0.25	1e-06
XDSO	[64]	0.25	1e-06
QDFA	[32]	0.25	1e-06
UFNW	[32]	0.1	1e-07
VZHH	[32]	0.1	1e-06
DLOB	[32, 16, 8]	0.25	1e-07
YSKV	[64]	0.1	1e-07
UTPG	[32]	0.25	1e-07
IWIG	[32]	0.5	1e-07
ZOVA	[64]	0.25	1e-07
QNRE	[6]	0.1	1e-07
FUUD	[6]	0.25	1e-07
WYBA	[6]	0.1	1e-06
XXIF	[64]	0.5	1e-07
PGSQ	[32, 16, 8]	0.5	1e-07
CODE	[4, 4, 4]	0.1	1e-06
QTGO	[4, 4, 4]	0.25	1e-06
EKHX	[6]	0.25	1e-06
ZDZT	[4, 4, 4]	0.5	1e-07
GXHS	[4, 4, 4]	0.1	1e-07
URYP	[6]	0.5	1e-07
JNHE	[4, 4, 4]	0.25	1e-07

Table 6: All configurations tested in the hyperparameter optimization for the ANN for visible accretion. The 'modelcode' is an internal code identifying a specific configuration. 'kernel_regularizer' is the amount of L2 regularization of ANN weights.