Using machine learning algorithms to enhance numerical weather prediction

Training random forest and neural networks with MEPS output to forecast hourly accumulated precipitation

UNIVERSITY OF BERGEN
GEOPHYSICAL INSTITUTE

Master thesis in Meteorology

2021/2022

Author: Inger Kristin Nesbø Gjøsæter

Supervisor: Asgeir Sorteberg, Geophysical Institute

Abstract

In this master thesis machine learning algorithms have been used in an effort to enhance numerical weather prediction. The learning algorithms used are random forest to provide a *no precipitation/precipitation* classification, and neural networks to predict the numerical value of *hourly accumulated precipitation amount*. Both algorithms has been trained on numerical model output, specifically output from the numerical model MEPS (MetCoOp ensemble prediction system) for two locations with distinctly different precipitation frequencies, Florida, Bergen and Blindern Oslo. After the data was classified by the random forest model, a neural network was trained for each individual class (i.e class 0: *no precipitation*, class 1: *precipitation*). The neural network show promising results according to the verification metrics, which suggest an improvement compared to MEPS. When splitting the forecast into categories of hourly accumulated precipitation amount (rain rates) the verification metrics show a good performance of no rain and light rain, however, it revealed a systematic underprediction of medium and heavy (hourly) rain rates at both locations. This underprediction is even more severe for Blindern, Oslo than for Florida, Bergen, illustrating the importance of enough data in machine learning. Though the algorithms was provided many samples, not enough of these samples actually meet the threshold of hourly rain rate $RR_{hourly} \geq 0.1 mmh^{-1}$, and thus there is not enough precipitation samples for the algorithms to capture the true distribution of the observed precipitation. MEPS also struggles to capture these hourly rain rates, though the numerical model manages to do so significantly better than the neural networks. Hence, the verification metrics favors the neural networks for $RR_{hourly} \in [0, 2.5\rangle\ mmh^{-1}$, and MEPS for $RR_{hourly} \in [2.5, \infty\rangle\ mmh^{-1}$. Visual inspections of plots of the predictions from the neural networks and MEPS, together with the observed precipitation show the same underprediction, which for Blindern is severe enough that visually one can deem MEPS a better fit to the observations. For Florida however, it is not clear which is better, and though the neural network is not obviously better by any means, it is very much comparable to MEPS concerning light rain ($RR_{hourly} \in [0, 2.5\rangle\ mmh^{-1}$), which in itself is an accomplishment that yield promise for future research.

Acknowledgments

First and foremost I would like to thank my supervisor Asgeir Sorteberg for guiding me through the process of writing this thesis and providing insightful comments. It has truly been an enjoyable process to work with this topic of machine learning and numerical weather prediction, not to mention a great learning experience. Additional thanks to Clio Michel and Jenny Sjåstad Hagen for assistance in programming and machine learning.

I would also like to thank my partner, family and friends for their continuous support and kind words, which has been really helpful through the ups and downs of this process. Thank you to my fellow students at GFI for the coffee breaks that uplift all the time spent at the reading hall.

Contents

List of Figures

List of Tables

Acronyms

NWP numerical weather prediction

MEPS MetCoOp ensemble prediction system

AI artificial intelligence

ML machine learning

RFC random forest classifier

ANN artificial neural network

RNN recurrent neural network

NN neural network

MLP multilayer perceptrons

RFECV recursive feature elimination with cross-validation

RFE recursive feature elimination

ROC Relative Operating Characteristic

ACC accuracy

POD probability of detection

POFD probability of false detection

FAR false alarm ratio

SR sucess ratio

CSI critical success index

SS skill score

MAE mean absolute error

Part I
Introduction

Today weather forecasts are a well integrated part of society, and is not only a tool used by governments and industries, but is readily available for the general population. For many people weather forecast is an important tool used to navigate their everyday lives, from choosing appropriate outerwear to plan outdoor activities and chores. Furthermore, many industries are heavily dependent on weather forecast in order to execute their operations, not only to prevent economic losses, but to prevent damage to the environment and to ensure safety for humans involved. In addition, weather forecasts are a crucial part of public safety, and is a tool that enables government agencies to take sufficient preliminary actions and precautions when faced with impending high-impact weather in order to both protect infrastructure and lessen the endangerment of human lives.
Today, numerical weather prediction (NWP) are computed several times a day by solving the governing equations numerically on supercomputers at an increasingly high resolution. However, at the start of the 20th century the predictability of the atmosphere was just getting recognized.

Vilhelm Bjerknes proposed the idea that it was possible to predict the future state of the atmosphere using the laws of physics and knowing the current state of the atmosphere. [Bjerknes, 1904] stated in his paper:

> *"If it is true as any scientist believes, that subsequent states of the atmosphere develop from preceding ones according to physical laws, one will agree that the necessary and sufficient conditions for a rational solution of the problem of meteorological prediction are of the following:*
>
> - *One has to know with sufficient accuracy the state of the atmosphere at a given time.*
>
> - *One has to know with sufficient accuracy the laws according to which one state of the atmosphere develops from another."*

Bjerknes' first point recognizes the need to observe current weather such that the future state of the atmosphere can be treated as an initial value problem, given that the equations governing how the atmosphere evolve in time and space are known (as stated in the second item). Though Bjerknes [1904] gave frameworks of what is needed to solve the system, both as far as the equations, initial conditions and potential implications, it was Lewis Fry Richardson in 1922 who proposed solving the system numerically and also showed how this could be done in practice [Kalney, 2006]. It took Richardson two years to complete the calculations manually and produce a six hour forecast of pressure. The forecast predicted a pressure change of 146 hPa when in reality there was little to no change, and his effort was at the time somewhat dismissed as optimistic. This overestimation of pressure change has in later years been attributed to the fact that the initial conditions contained fast-moving gravity waves which obscured the meteorological signal [Kalney, 2006]. Although Richardson's forecast was incorrect, his work illustrated how to solve the system proposed by Bjerknes, and was an important key in the further developments in the field of numerical weather prediction. After further development in dynamical meteorology, Charney et al. [1950] used one of the first electronic computers, ENIAC, to compute a 24 hour forecast of pressure change at 500 hPa yielding promising results, and in September 1954 Sweden deployed the first *operational* numerical weather prediction (NWP) system, followed by the US six months later [Kimura, 2002, Kalney, 2006].

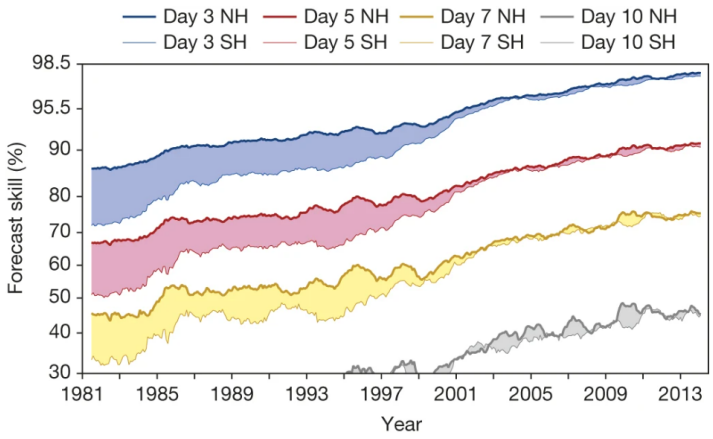


Figure 1: The evolution of forecast skill from 1981-2015, for $3-, 5-, 7-$ and $10-$day forecasts of the height of the 500 hPa pressure surface [Bauer et al., 2015]. The thicker line shows the forecast skill for the Northern Hemisphere (NH) and the thinner line represents the Southern Hemisphere (SH). 1999 marks a clear shift in the forecast skill before and after, due to the improved utilization of satellite data.

The performance of numerical models have improved steadily since the first operational NWP was launched in 1954, and this can be attributed to several reasons. This improvement is closely related to the exceptional increase in computational power since the 1980s, which has made it possible to operate with increasingly higher resolution. Higher resolution mean we can resolve more processea within the grid, hence fewer parameterizations are needed. Resolution strongly affects model performance, and usually higher resolution leads to better model performance [Kalney, 2006].
In addition, the observational network needed to provide data about the current state of the atmosphere has been greatly expanded worldwide, and improved methods of data assimilation has resulted in initial conditions that resemble the actual state of the atmosphere more closely. This is important because, as discovered by Edward Lorenz in the 1960s, the atmosphere is chaotic, meaning that small changes in initial conditions can grow into vastly different systems in a relatively short temporal span.
Figure 1 shows the evolution of forecast skill from 1981 until 2015, for the height of the 500 hPa pressure surface [Bauer et al., 2015]. The $3-, 5, 7-$ and $10-$day forecasts are plotted in blue, red yellow and gray respectively, and the thicker curve represents the skill for the Northern Hemisphere (NH), while the thinner curve represents the Southern Hemisphere (SH). Prior to 1999 there is a significant difference in forecast skill between the two hemispheres. Although this gap was slowly getting smaller in the years prior as well, the forecast skill in SH quickly improved after 1999. This can be attributed to a combination of increasing satellite coverage and better utilization of the massive amounts of data provided by these satellites, and as evident in fig. 1, the forecast skill is today comparable between the hemispheres.

Even with finer resolution there is still sub-grid processes we need to parameterize in order to represent in the numerical models. Precipitation is formed by microphysical processes which happens on a *much* smaller spatial scale than the model resolution, and thus these microphysical processes needs to be parameterized. Additionally, difference in local topography within a grid cell results in difficulty predicting certain variables, such as precipitation which is enhanced by orographic effects. Simplified microphysics and smoothed topography that remain unresolved are why it is particularly difficult to forecast precipitation, not only when and where it will precipitate, but particularly how much precipitation will fall. Convective precipitation, which form from ascending air which has been heated by the adjacent surface and is common during summer months, often result in local rain showers that is difficult to forecast when and where ahead of time. Frogner et al. [2019] researched the predictability of MEPS forecast of hourly

precipitation and found it was greatly reduced for scales smaller than 65km after the 6-h forecast lead time.
As mentioned initially, weather forecasts are an important tool in today society, and this also holds true for precipitation forecasts specifically. As for public safety and infrastructure it is detrimental to forecast heavy, and especially extreme precipitation events sufficiently, as it can result in land slides and flooding, potentially causing loss of lives and severe economic damage. Furthermore, many industries need accurate forecast to execute their operations such as the energy sector, especially the renewable. The hydro power industry for example, provide the largest source of energy in Norway and constantly have to make decisions about production based on precipitation forecasts. Other industries, like aviation, the oil industry and other maritime industries also rely heavily on accurate forecasts in general to ensure safety and prevent significant economic risk. Therefore, the work on further improving numerical weather prediction is ongoing, and studies integrating machine learning (ML) into the process of NWP has increases over the last years following this pursuit.

While numerical models bases the predictions on prognostic and deterministic equation, as well as assumptions and initial conditions, machine learning algorithms produce output based on the data itself by identifying patterns and relationship within the data provided in the training process [Gagne et al., 2017]. Machine learning (ML) is an interdisciplinary field, based on concept from mathematics, statistics, information theory and computer science, to mention some, and is not a new concept. It is based on similar statistical methods as the post-processing techniques used in NWP for decades to produce the final forecast that is presented to the users, by reducing biases and make the output more representative for local conditions [Kalney, 2006]. However, these statistical methods have to be manually programmed, while machine learning algorithms are able to *learn* for the data, which means, following Mitchell [1997] definition, the performance of the ML algorithm is improved by experience, and thus does not need to be explicitly programmed (this will be further explained in sec. 4). Another strength of ML algorithms is that they can process large amounts of data quite quickly compared to traditional methods, in addition to the ability to identify complex patterns and relationship that are unknown to humans at the time [McGovern et al., 2017].

The concept of learning machines has intrigued humans since the invention of computers, and the term *machine learning* was officially introduced by Samuel [1959]. As mentioned previously, machine learning is based on many concepts making it very much interdisciplinary and is a sub-branch of the field of artificial intelligence (AI), which tries to emulate the processes in the brain to simulate intelligence, logic reasoning and thinking in machines, just as seen in humans. The field of AI has seen dramatic development in the last decades, and the majority of this can be attributed to the developments within machine learning in particular [Goodfellow et al., 2016]. One order of magnitude increase in computing power every five years since 1980, with the massive amounts of data that has become available has contributed to the significant breakthroughs, yielding promising results in many fields. This has resulted in an increasing interest and popularity in the last decade, and many other fields has also realized what a powerful tool ML can be, including the field of numerical weather prediction.

An increasing number of studies have been conducted trying to implement machine learning into the process of numerical weather prediction. McGovern et al. [2017] demonstrated how machine learning can be utilized in the predictions of different high-impact weather phenomena. They used a random forest (RF) to classify different precipitation types and found a significant improvement from the NWP model in the study, particularly for the minority classes of ice pellets and freezing rain. Another study using random forest was conducted by Ahijevych et al. [2016], whom tried to forecast the probability of initiation of a mesoscale convective system (MCS-I).

The RF model was able to detect 99% of the initiation events, however, this was accompanied by an significantly high false alarm rate because the model either detected the initiation prematurely, or was unable to distinguish between an ongoing MCS-I event, and the initiation of a new event. Gagne et al. [2017] used a combination of random forest and a regression model to first classify a storm as either hail-producing or not, followed by predicting the hail size. They found random forest to performed rather well at the classification part, while the regression model struggled to predict hail size with sufficient accuracy. When using RF to improve radar-base precipitation nowcasting, Mao and Sorteberg [2020] found overall improvements in the verification metrics of the RF model compared to the NWP and radar nowcasts in the study.

The objective of this thesis share similarities with the studies just presented, namely to research whether machine learning methods can be used to enhance numerical weather prediction (NWP), specifically predictions of *hourly accumulated precipitation amount*. The machine learning (ML) algorithms used are random forest and neural network, with both algorithms trained on data from AROME-MetCoOp ensemble prediction system (MEPS). This study can be separated into three distinct parts; feature selection, classification by random forest and regression using neural networks. The random forest provide a classification of *no precipitation/precipitation*, and two separate neural networks are trained (one for each class). After the random forest has classified all the data, the data is then passed to the corresponding neural networks predicting the *hourly accumulated precipitation amount*. This procedure is tested on data from two different location with distinctly different precipitation frequency; Florida, Bergen and Blindern, Oslo. In addition, each location have several datasets with various sets of variables and forecast lead times. The full overview of the different datasets used to train the ML models are provided in sec. 5.
The research questions answered in this thesis is the following:

- How does the performance of the ML algorithms change with different forecast lead time intervals and features available?
- Will training two different neural networks, one for samples classified as precipitation and one for samples classified as no precipitation, lead the neural network (NN) to predict fewer false negatives and false positives (i.e precipitation classified as no precipitation, and no precipitation classified as precipitation)?
- How does the ML algorithms predictions compare to MEPS, and is it possible to improve forecast skill with ML?
- How does the performance of the ML models differ between two locations with distinctly different precipitation frequency?

The main text is structured in the following way; Part II provides theory of relevant concept such as cloud formation and precipitation, numerical weather prediction, exploratory data analysis and machine learning. In Part III the methods used in this study is presented, as well as the data used. Results and discussions are presented in Part IV, with the different section presenting the findings of feature selection, random forest and neural network. Because these are subsequent steps, it is necessary to discuss and conclude the findings after each step. Lastly, the conclusion and further work is presented in Part V.

Part II
Theory

1 Cloud formation and precipitation

In order to have precipitation, cloud formation must first occur. Clouds form in four ways; convective, convergence, frontal and orographic lifting.

1.1 Orographic lifting

Orographic lifting, illustrated in fig. 2, occurs when an air parcel approaches orographic features (i.e mountains etc), and on the windward side is forced upwards along the mountain side, thus cooling the parcel adiabatically by expansion, and once the air parcel reaches the lifting condensation level (LCL), it continues to cool following the pseudoadiabatic lapse rate, a cloud starts forming and precipitation can occur. This is called orographic precipitation, and occurs on the windward side of the mountain as the air parcel ascend over the mountain, permanently removing moisture from the parcel. On the downwind side, the parcel moves back down the mountain side. Because of the lower water content from the precipitation, as the parcel is compressed during the descend, it warms following the dry adiabatic lapse rate. The dry adiabatic lapse rate is greater than the pseudoadiabatic lapse rate, and thus the temperature of the air parcel as it descends exceeds the original temperature of the parcel before it ascended over the mountain on the windward side [Hakim and Patoux, 2018].

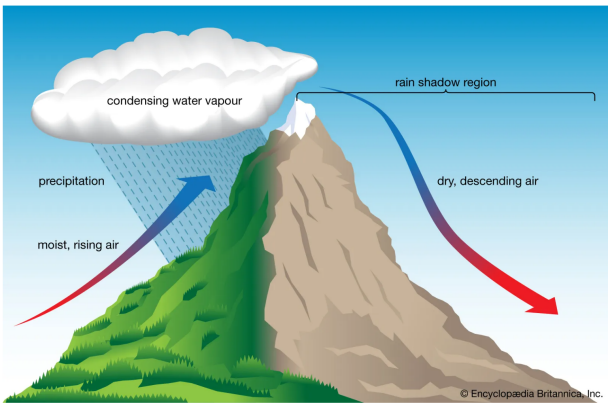


Figure 2: A parcel of moist air is forced by orographic lifting up the slope of the mountain, creating clouds and precipitation on the windward side and a heating of the air on the leeward side (see text for further details). Figure from Britannica.

1.2 Convective lifting

A process which lead to many weather phenomena is convection. As shown in fig. 3a, the air parcel is heated by the surface, either from the surface adjacent to the parcel being heated by solar radiation, or by the parcel itself moving over a warm surface. Because warm air has a lower density, the parcel begins to rise, and at the LCL condensation begins and a cumulus cloud is formed. The stability of the atmosphere will determine how far vertical reach the cumulus cloud will have [Ahrens and Henson, 2018]. The farther vertical extent the unstable layer has, the taller the cumulus can become.

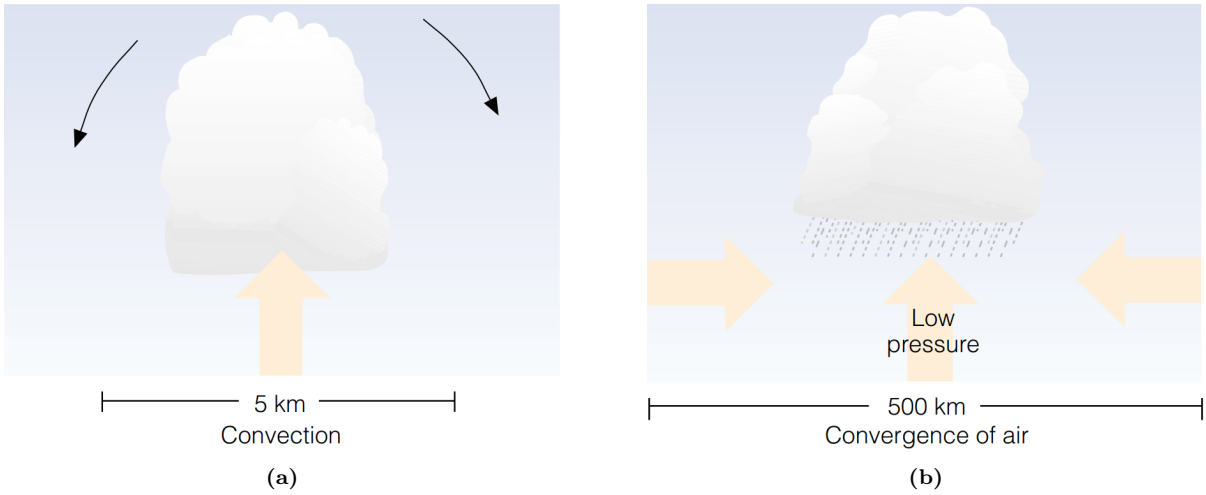


Figure 3: Figure illustrates 3a convective lifting and 3b lifting by convergence. Figures from Ahrens and Henson [2018]

1.3 Lifting by convergence

Another way clouds can form is from the lifting of air due to convergence. Convergence is when air masses meet at a certain point in space, forcing some of the air to move upward (fig. 3b). This allows for adiabatic cooling and cloud formation once saturation has been reached, and in some instances, precipitation. This can typically occur when air masses flow over a body of water and then reaches land. The air that reaches the land first will slow down due to the increased friction of the surface. The air still moving over the body of water will continue with the same speed, and at the boundary between land and water there will be a convergence as faster moving air catches up to the slower moving air [Hakim and Patoux, 2018].

1.4 Frontal lifting

A front is the boundary between two air masses with different densities, mainly due to a noticeable difference in temperature (and moisture content). There are two main fronts; warm front and cold front. This is usually seen as part of a low pressure system.

A warm front is a warmer air mass moving towards colder air. Because the warmer air has a lower density than the colder air, the warmer air will be forced to ascend over the colder air. This ascending air produce nimbostratus clouds and precipitation, and because the warmer air moves over the colder air (see fig. 4a), the cloud formation and precipitation occur *prior* to the passing of the warm front.

A cold front on the other hand, consist of a colder, and thus denser air mass, and force the warmer air mass to rise over the air mass of the cold front. As seen in a warm front, the rising of the warmer air mass creates nimbostratus and precipitation, but in contrast to the warm front, this occurs *behind* the cold front (see fig. 4b).
As mentioned, these fronts are typically a part of a low pressure system where the warm front is ahead of the cold front (fig. 6b). Because the cold front moves slightly faster than the warm front, the cold front eventually catches up with the warm front, and pushes this air upwards, until all of the warmer air mass sits above the colder air mass [Strangeway, 2007], creating what is called an *occluded front*.

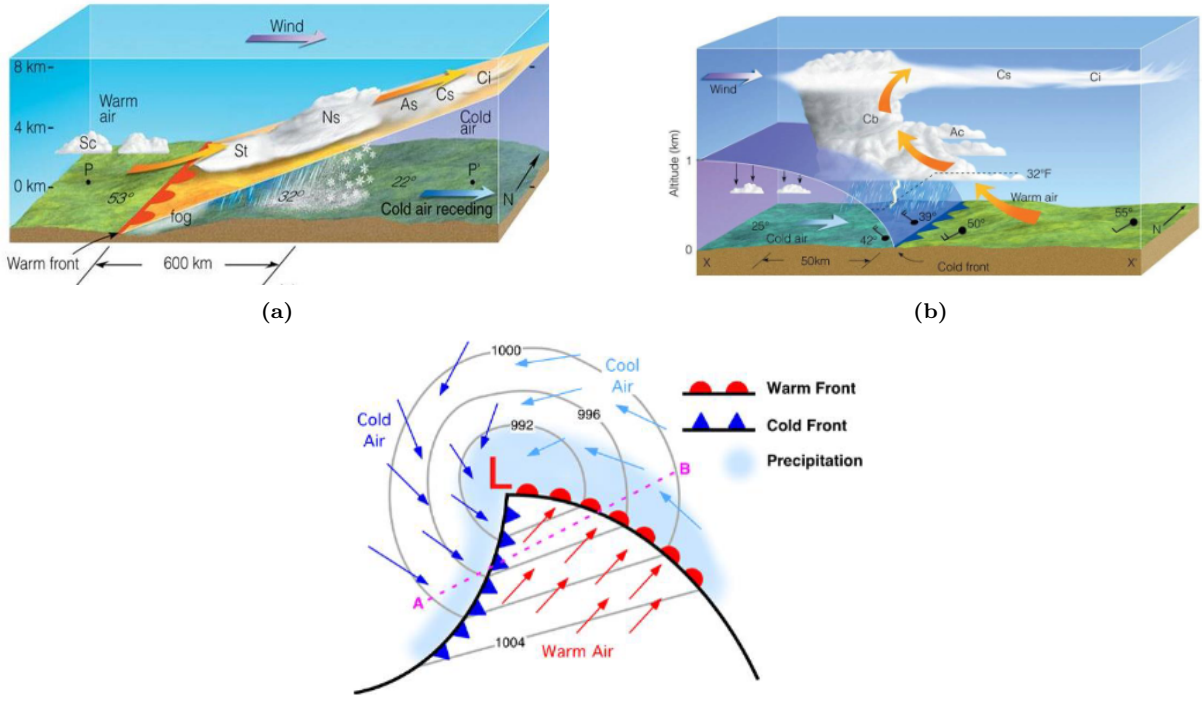


Figure 4: Figures show the characteristics of 4a a warm front and 4b a cold front. In the case of a warm front, the precipitation occurs ahead of the approaching front, and in the case of a cold front, the precipitation occur behind the front. Figures from Ahrens and Henson [2018]. Fig. 6b show a typical developing synoptic low pressure system (cyclonic circulation), with the cold front behind the warm front, and the accompanying precipitation band. Fig. from Wallace and Hobbs [2006].

2 Numerical weather prediction

In this section different aspects of numerical weather prediction will be presented, from the governing equation and how they can be numerically solved, to data assimilation and ensemble predicting.

2.1 The governing equations

Below is the governing equations, often referred to as *Euler equations* when not containing friction $\mathbf{F}$ [Kalney, 2006]: The momentum equation defined as

$$\frac{D\mathbf{u}}{Dt} = -\frac{1}{\rho} \bigtriangledown p - \bigtriangledown\phi - 2\Omega \times \mathbf{u} + \mathbf{F} \tag{1}$$

where $\frac{D}{Dt} = \frac{\partial}{\partial t} + u\frac{\partial}{\partial x} + v\frac{\partial}{\partial y} + w\frac{\partial}{\partial z}$, $\mathbf{u} = (u, v, w)$ is the three wind-components, and the first term on the right hand side is the pressure gradient force, second term is the gravitational force, third is the Coriolis force and last term is friction.
The continuity equation defined as

$$\frac{\partial \rho}{\partial t} = - \bigtriangledown (\rho \mathbf{u}) \tag{2}$$

where ρ is density and $\mathbf{u}$ is the wind vector.
The state equation for an ideal gas is given by

$$p = \rho R T \tag{3}$$

where p is pressure, ρ is density, T is temperature and R is the gas constant for air.
The conservation of energy equation is defined as

$$Q = C_p \frac{DT}{Dt} - \frac{1}{\rho}\frac{Dp}{Dt} \tag{4}$$

where C_p is the specific heat at constant pressure. The conservation of moisture is defined as

$$\frac{\partial \rho q}{\partial t} = - \bigtriangledown (\rho \mathbf{u} q) + \rho(E - C) \tag{5}$$

where q is the mixing ratio and E and C represent evaporation and condensation respectively.

2.2 Discretization of the governing equations

Because the governing equations cannot be solved analytically, it is necessary to discretize the equations by finite-difference approximations of space and time in order to solve them numerically [Warner, 2011]. To illustrate this, consider defining discrete values for x and t as $x_j = j\Delta x, t_n = n\Delta t$, the advection equation $\frac{\partial u}{\partial t} = -c\frac{\partial u}{\partial x}$ can be approximated by the finite-difference equation

$$\frac{U_j^{n+1} - U_j^n}{\Delta t} + c\frac{U_j^n - U_{j-1}^n}{\Delta x} = 0, \tag{6}$$

which is referred to as an *upstream scheme*, and for which both the spatial and temporal discretization is centered around the point $(j\Delta x, n\Delta t)$ [Kalney, 2006]. The upstream scheme is just one of many possible ways to discretize the governing equations.

2.3 Parameterization

Due to the discretization used to solve the equations numerically, some processes will be *unresolved*, meaning these processes occur on a smaller scale than the Δx used in the discretization. However, these subgrid-scale processes are still important in weather prediction and their effects need to be included in the model. This is achieved by parameterizations, i.e describing their effects in the form of resolved processes instead. Figure 5 shows a schematic from Bauer et al. [2015] of subgrid-scale processes that are important for weather prediction and need to be parameterized. These subgrid-scale processes include, among others, molecular-scale processes such as condensation, evaporation and all other microphysical processes involved in cloud formation, and turbulent mixing in the boundary layer.

2.4 Data assimilation

In order to solve the equations presented above, which is an initial-value/boundary-condition problem, one need to know the initial state of the atmosphere as accurately as possible. This initial state is referred to as the *analysis*. The analysis is based on a short-range forecast and observation, combined statistically using *data assimilation* [Kalney, 2006]. Data assimilation is defined by Talagrand [1997] as "the process through which *all the available information is used in order to estimate as accurately as possible* the state of the atmospheric or oceanic flow".
As mentioned, fig. 1 in Part I show the evolution of forecast skill from 1981 to 2015, with the thicker line represents the forecast skill for Northern Hemisphere while the thinner represents the Southern Hemisphere. Prior to 1999 there was a significant difference in skill between the two hemispheres, however, after 1999 this difference is greatly reduced. This is due to the better methods of data assimilation, as well as better use of satellite data in the process of data assimilation [Bauer et al., 2015].

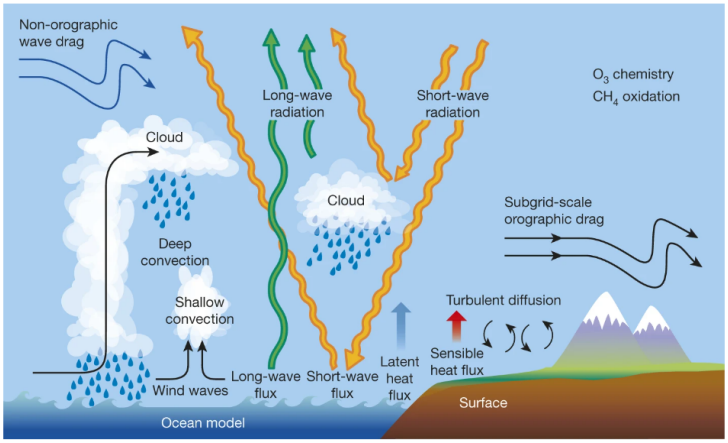


Figure 5: A schematic presentation of unresolved processes which is represented by parameterization i numerical weather prediction [Bauer et al., 2015].

2.5 Ensemble forecast

Traditional statistical methods is not sufficient to represent the uncertainty in NWP. Chaos theory of Edward Lorenz in the 1960s showed how small perturbations in the initial conditions can over time evolve into distinctly different trajectories (as illustrated by the Lorenz attractor in fig. 6), and because of this, even with a perfect model and close-to perfect initial conditions, there is a theoretical limit to the predictability of the atmosphere. In an effort to account for this inherent uncertainty, ensemble forecasts are used. The *base member* in the ensemble (red line in fig. 6a) is the numerical model run on the analysis provided by data assimilation. The various *ensemble members* (blue lines in fig. 6a) are model runs which have been perturbed in one way or another, for example by perturbing the initial conditions. The incentive of ensemble forecasting is to capture the true trajectory of the observed system in the spread of the ensemble members (gray area in fig. 6a).

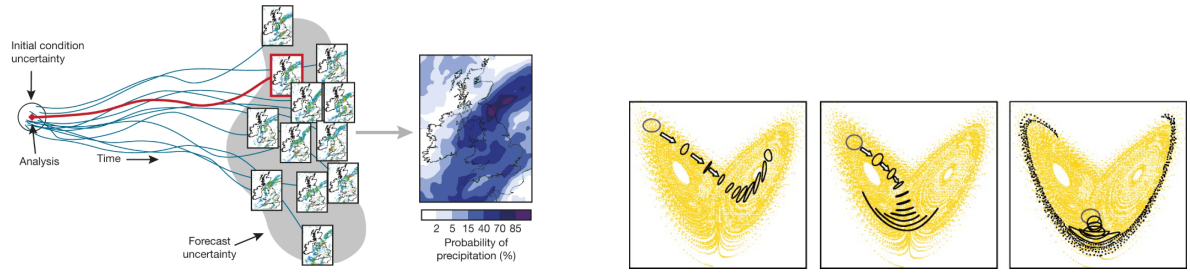


Figure 6: Figure 6a illustrates ensemble prediction. The red line marks the *control*, and the blue lines marks the different members of the ensemble which is perturbed in some way. The lines show the temporal evolution of the forecasts, illustrating how the different forecasts diverge as time increases. The gray-shaded area symbolizes the forecast uncertainty, and the objective is to encompass the true event in the range of the ensemble. The left figure shows how this ensemble prediction result in a probabilistic forecast of precipitation. [Bauer et al., 2015]. Figure 6 is a schematic of a Lorenz attractor, which illustrate how different initial conditions have different the uncertainty, and might end up taking vastly different trajectories.

2.6 Forecast verification

In order to assess the usefulness of a forecast, it is necessary to define what constitutes a *good forecast*. This can be somewhat challenging to define, as it can be quite subjective and one forecaster's assessment of *goodness* might not agree with another, or perhaps even more com-

monly, not reflect the end-users experience. In his attempt to define a clear definition of what constitutes a good forecast, Murphy [1992] defined three different types of goodness. The first type, termed *consistency*, refer to the agreement between the forecast and the forecaster's better judgment, meaning the forecast needs to align with the forecaster's internal knowledge base. The second type of goodness require the forecast to be in agreement with the matching observations, and is termed as the *quality* of the forecast. Lastly, the forecast needs to be of *value*. This third type of goodness takes the end-user's experience of the forecast into consideration, and require that the forecast must provide decision-makers with information providing economic or other likewise societal benefits. Murphy [1992] argued that without a sufficient definition of goodness, the actual usefulness of the forecast is undermined.

Forecast verification provide a quantification of how well the forecast corresponds with the actual observations, and thus quantify *type 2 goodness*, i.e the *quality* of the forecast. The verification measures used is dependent of the type of forecast. For deterministic forecasts of discrete variables (i.e yes/no forecast) *accuracy* is commonly used, which measures how many times the forecast predicted true positive compared to total number of positive instances (instances of yes).
Deterministic forecasts of continuous variables on the other hand, uses verification measures that provide information about how close the predicted numeric value is to the observed value, for example the *mean absolute error*. For both forecasts of discrete and numeric values it is useful to give an estimate of how well the forecast generally fit the observations, referred to as the forecast's *mean error* or *bias*. Another verification measure used in both types of forecasts is *skill*. Skill is a measurement of the forecast's performance relative to some reference forecast, and is a measure that is particularly useful when comparing different models.

The high-resolution of today's models have resulted in remarkable improvements of the forecast of certain variables, however, this high-resolution also poses some challenges in the verification process. The high-resolution of today's models can provide detailed information about the spatial distribution of the variable field, and though this looks very realistic, it can be difficult to verify whether or not the model is correct because details about the actual observed variable is often not available at such a high resolution. In addition, traditional verification is also done in a point-to-point comparison between prediction and observation, often resulting in poor verification scores that does not reflect the true quality of the forecast [Gilleland et al., 2009]. There are new verification methods developed to account for this, namely filtering methods and displacement method, but this will not be further explained as this thesis utilizes data of one grid cell only, while these methods require a variable *field*.

3 Exploratory data analysis

Feature engineering and selection is perhaps the most important step in the machine learning process. As a part of this process, exploratory data analysis was done to get a better insight of the relationships within the dataset, and also between the dataset and the observations.

3.1 Person correlation and Spearman rank correlation

In order to explore the relationship between two variables, x and y, one can look at the *Pearson correlation coefficient*, which is defined by

$$r_{xy} = \frac{\frac{1}{n-1}\sum_{i=1}^{n}[(x_i - \overline{x})(y_i - \overline{y})]}{[\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x})^2]^{\frac{1}{2}}[\frac{1}{n-1}\sum_{i=1}^{n}(y_i - \overline{y})^2]^{\frac{1}{2}}}, r_{xy} \in [-1, -1] \tag{7}$$

where the numerator can be interpreted as the sample covariance of both variables over the product of the standard deviation of both variables Wilks [2011]. A correlation of 1 means a perfect positive linear association between the variables, i.e when one variable increases, the other increase with an equal amount. In contrast, a correlation of -1 means a perfect, but negative, linear association, meaning when one variable increases, the other decreases with an equal amount (of absolute value). Two variables with no linear relationship would have a correlation of 0.
One weakness of the Pearson correlation is that it might underestimate the relationship between two variables if the relationship is not entirely linear. In these situations *the Spearman rank correlation coefficient* would be better to use, which is calculated using the rank for the data pair (x_i, y_i) (instead of the actual value) in equation 7. In other words, the data pair is transformed from (x_i, y_i) where $x_i \in [min(x), max(x)], y_i \in [min(y), max(y)]$ to $(rank_{x_i}, rank_{y_i}) \in [1, n]$. Because the rank of a specific data point is an integer between $[1, n]$, the average of the rank is $(1 + n)/2$ and the standard deviation $n(n^2 - 1/[12(n - 1)])$ [Wilks, 2011], the Spearman rank correlation can be found using

$$r_{rank} = 1 - \frac{6 \sum_{i=1}^{n} D_n^2}{n(n^2 - 1)} \tag{8}$$

where D_i is the difference in rank between the two variables. Like the Pearson correlation, the Spearman rank is also bounded by -1 and 1. However, though a Spearman rank of 1 also means a perfect relationship between the two variables, it does not mean they will both increase with an equal amount (as Pearson correlation indicates).

3.2 The correlation matrix

The statistical measures described above can only tell us something about the relationship between *two* variables. In order to explore the relationship between ≥ 3) variables, other methods must be utilized. If you have a dataset with K variables, where $K \in [3, \infty\rangle$, *The correlation matrix* show the correlations between all distinct parings of these K variables [Wilks, 2011], as illustrated below. The matrix can be calculated using either the Pearson correlation or the Spearman rank. The diagonal elements $r_{1,1} \dots r_{K,K} = 1$ because this is the correlation of the variables themselves. The matrix is symmetric, meaning $r_{1,2} = r_{2,1}$, and we therefore only have $K(K - 1)/2$ distinct pairings of these K variables.

$$R = \begin{bmatrix} r_{1,1} & r_{1,2} & \dots & r_{1,j} \\ r_{2,1} & r_{2,2} & \dots & r_{2,j} \\ \vdots & \vdots & \ddots & \vdots \\ r_{i,1} & \dots & \dots & r_{i,j} \end{bmatrix} \tag{9}$$

3.3 Hierarchical clustering

Another method of exploratory data analysis is *cluster analysis*, where we try to divide the data into smaller groups of unknown identities. One way to do this is by hierarchical clustering. This is essentially done by first pairing the two variables closest together, then merging this group together with another existing group. In more general terms, if we have a dataset with n variables, the data will be divided into $n - 1$ groups at the initial step in the clustering. At the next step, two of the existing groups will be merged, resulting in $n - 2$ groups. This process is continued until two groups remain [Wilks, 2011]. The hierarchical clustering will hopefully result in an informative split of the data at some of the intermediate steps.
In this thesis, *Ward's minimum variance method* was used to decide which pair of groups to merge at each step. Consider $G + 1$ groups. The next step in the clustering is to merge the two

groups which minimizes

$$W = \sum_{g=1}^{G} \sum_{i=1}^{n_g} ||\mathbf{x}_i - \overline{\mathbf{x}}_g||^2 = \sum_{g=1}^{G} \sum_{i=1}^{n_g} \sum_{k=1}^{K} (x_{i,k} - \overline{x}_{g,k})^2 \tag{10}$$

In order to find the best merge, equation 10 needs to be calculated for all possible pairs of existing groups. This process is recursively done until the data is divided into two groups. To illustrate the hierarchical clusters found by Ward's method one can use a Dendrogram, as illustrated in fig. 7.

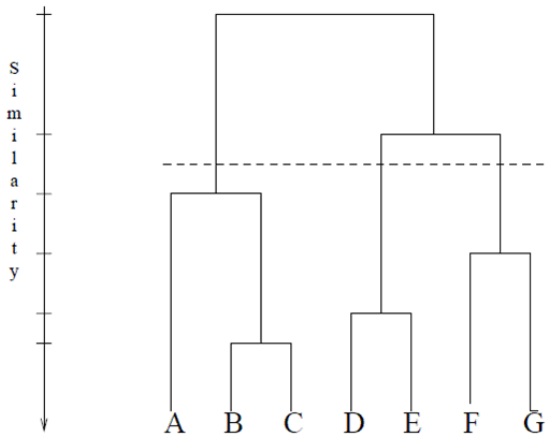


Figure 7: Schematic of a Dendrogram. At each step, two groups are merged. In this figure, at the start there are 7 groups (A-G), and at the end there are only two. [Joseph, 2018].

4 Machine Learning

In this section a brief introduction about machine learning is presented. Machine learning is as mentioned a sub-branch of artificial intelligence.
In traditional rule-based systems one must specify a manually defined program and input in order for the computer to return an output, whilst in machine learning one specifies an input and output and then the computer learns (and returns) the program itself (fig. 11). In other words, the machine learning algorithm tries to learn patterns between the input and output, to be able to predict future situations based on historic data. This is called *inductive inference*, and is in the field of machine learning referred to as the learning algorithms ability to *generalize*. But what does it mean for an algorithm to learn? Mitchell [1997] provided the following definition of learning:

> A computer program is said to learn from **experience E** with respect to some class of **tasks T** and **performance measure P**, if its performance at tasks in **T**, as measured by **P**, improves with experience **E**.
> - Mitchell [1997]

The following sections will specify the terms used in this definition further.

4.1 Task, T: The purpose of the learning

What is the purpose of the machine learning algorithm, what need is it set to fulfill? The task is not the learning itself. The learning is simply the mean to solve the task at hand. In this thesis the task is two-fold, first to classify the data into *no precipitation/precipitation*, then to predict a numerical value of *hourly accumulated precipitation amount* (for simplicity also referred to as hourly precipitation amount). These are examples of two different classes of tasks,

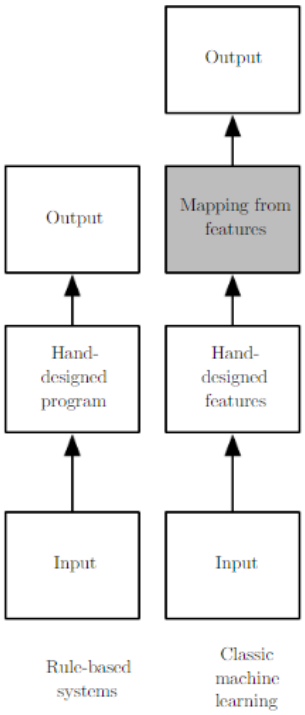


Figure 8: Diagram of machine learning versus traditional rule-based systems [Goodfellow et al., 2016]. Here, a traditional rule-based system is a program that has rules explicitly stated by a developer and from this predicts an outcome, while machine learning *learns* these rules from the data itself.

namely *classification* and *regression* respectively. There are many different tasks one can utilize machine learning to solve, such as anomaly detection, language translation and even generating new samples similar to the provided training data (synthesis). Only classification and regression will be further explained, as these are the classes of tasks to be solved in this thesis.

4.1.1 Classification

In classification tasks the algorithm is set to assign each sample to one of k classes, also often referred to as a label (these terms will be used interchangeably). In this thesis specifically the classification task is to classify the sample as either *no precipitation* or *precipitation*. In cases like this where there are only two classes it is common to use binary labels such as $[0, 1]$ or $[-1, 1]$, referring to *no precipitation* or *precipitation* respectively [Goodfellow et al., 2016].
Usually in such classification tasks with k classes, the algorithm has to approximate a function $f : \mathbb{R}^n \to 1, .., k$, such that it is able to predict class $\mathbf{y}$ from input vector $\mathbf{x}$, satisfying

$$\mathbf{y} = f(\mathbf{x}) \tag{11}$$

It is worth mentioning that there are different subtypes of classification tasks, and in this particular classification task, the output is *one* class (of k classes). In other classification tasks the algorithms may provide a probability distribution of the classes instead of one deterministic class [Goodfellow et al., 2016].

4.1.2 Regression

Regression tasks are quite similar to classification, however, instead of predicting the class of the sample, the algorithm has to provide a numerical value [Goodfellow et al., 2016]. Just as

in classification, the task is to learn a function such that $\mathbf{y} = f(\mathbf{x})$, but in this case the desired output $\mathbf{y}$ is a numerical value, and not a binary one, meaning $f : \mathbb{R}^n \to \mathbb{R}$.
In this thesis in particular, the regression task is to provide the numerical value of *hourly accumulated precipitation amount*.

4.2 Performance, P: How to evaluate learning

The *performance* is the evaluation of the algorithm's ability to solve the task. The performance measure must be appropriate for the task (for example accuracy for classification or mean absolute error for regression).

4.2.1 Generalization error

The performance measure P calculated during the training process is referred to as the *training error*. However, the training error does not tell us anything about how well the algorithm is able to *generalize*, only how well it is able to fit the training data. In order to assess the algorithm's ability to generalize, new data (not seen in training) is needed to provide an unbiased evaluation of the model's performance. The performance measure P on the previously unseen data (referred to as the *test* data) is called *generalization error*. When talking about evaluating the model performance it is the generalization error that is referenced.

4.2.2 Overfitting and underfitting

The difference in *training error* and *generalization error* gives us valuable information about how well the algorithm is able to learn. If the training error is low, but the generalization error is significantly higher, the learning has resulted in *overfitting*. This means that the algorithm has fit so well to the training data that it has not only learned the real patterns within the data, but also the noise. Thus the algorithm is fit too closely to the training data, and generalizes poorly.
On the other hand, if the training error is also high, we have a situation of *underfitting* the data. This means that the algorithm is unable to learn efficiently, usually either because the algorithm is too simple for the complexity of the task or the algorithm is not provided with enough training samples to ensure learning.

4.3 *Experience*, E: Different types of learning

The experience the algorithm is using to learn from is the data itself, where each sample is a learning experience. Variables within the dataset used in the training is referred to as predictor (stems from statistics) or features, and is used interchangeably. The data used as experience is referred to as the *training data*.

There are different ways the algorithm can experience the data, and thus learn. There are mainly three ways of learning; supervised learning, unsupervised learning and reinforcement learning. The learning style which is appropriate to use depends on the nature of task T.

4.3.1 Supervised versus unsupervised learning

In supervised learning the developer provide both input and output, and the learning algorithm has to map a function (also referred to as a program) based on the provided input-output pairs.

The end goal of supervised learning is to predict the next output when presented with new, unseen input. The learning style utilized in solving both the task of classification and regression in this thesis is supervised learning.

With unsupervised learning on the other hand, the developer provides the algorithm with input only, and then it is up to the algorithm to make sense of the data. The task in unsupervised learning is typically to find similarities between data and split the data into different clusters based on these similarities.

Though these are distinct learning styles with supervised learning being more task driven and unsupervised learning being more data driven, the line between them is blurry, and often the learning can be a mix of both [Goodfellow et al., 2016].

4.3.2 Reinforcement learning

In reinforcement learning the algorithm has to learn by trial and error in an interactive environment without any input from a developer. An example of this is an AI (artificial intelligence) learning to play a videogame without any initial guidelines of how the videogame works or what the objective of the game is, but by trial and error from repeatedly running through the game the algorithm will learn this. In other words, the algorithm learns by making mistakes.

4.4 Machine learning algorithms

4.4.1 Decision tree

Generally, a decision tree recursively partition the feature space based on certain thresholds, resulting in a set of rectangles [Ruíz et al., 2015]. There are different decision tree algorithms, and CART and C4.5 is presented here. The essence of both is to search the feature space and select the feature and associated threshold resulting in the highest information gain. This process is recursively done until the stop-splitting criteria is met, which often is either the maximum depth of the tree or minimal number of samples within the node (these criteria is decided by the developer through defining the hyperparameters) [Kuhn and Johnson, 2003]. In other words, the objective is to select the feature to split that would result in the biggest decrease of *node impurity* [Breiman et al., 1984]. *Node impurity* refers to the proportions of the two classes within a node, with the highest impurity when there is an equal proportion of both classes, and the smallest when there is only one class within the node (i.e the node is pure).

In order to assess which split would provide the highest information gain, and thus decrease node impurity, it is necessary to implement a *gain measure*. In cases with two classes, p corresponds to the proportion of the second class and $c(p)$ corresponds to the implemented gain measure. Two gain measures for classification is the Gini index and information gain, the former defined as

$$C(p) = 2p(1-p) \tag{12}$$

and *information gain(split) = entropy(before split) - entropy(after split)*, with entropy defined as

$$C(p) = -p\,log\,p - (1-p)\,log\,(1-p) \tag{13}$$

The Gini index and information gain is the gain measures used by CART and C4.5 respectively [Ruíz et al., 2015, Shalev-Shwartz and Ben-David, 2014].

A decision tree is built from the top with a *root node*, which branches into either *internal nodes* or *leaf nodes*. An *internal node* (often also referred to as *child node* or *decision node*) is, like the root node, a test of a selected feature and each branch from this node corresponds to a possible value of said feature (decided by the threshold). A leaf node provides the final prediction. A sample will be passed down the tree, the path decided by the values of it's specific features, and proceed further down the tree until it reaches a leaf node (i.e gets classified) [Mitchell, 1997]. Figure 9 is a an example of a simple decision tree for playing an unspecified sport activity, where the task is to classify whether the weather is suitable to do the unspecified activity or not (P: positive instance, N: negative instance)[Quinlan, 1986]. The features in this example are 'outlook', 'humidity' and 'windy', and the possible values they can have is discrete. In this example the *root node* is 'outlook', which has three branches (one for each possible value of 'outlook'). *outlook = overcast* leads directly to the *leaf node* of *Positive instance (P)*, while *sunny* and *rain* are *internal nodes*, and each provide a root node for its own *subtree*.

In the example presented above the possible values of each node is discrete, but this method can also be used for numerical features. In these cases there will be two branches from each root/internal node, and the sample will either move left or right (branch) depending on whether the value of the sample is above or below a certain *threshold* of the numerical feature.

There are several advantages to such tree structured methods. To mention some, the trees are not very affected by outliers or misclassifications in the learning data, in addition to their output being easily interpreted and their structure intuitive to follow [Breiman et al., 1984]. However, there are also some weaknesses to the algorithm. One of these weaknesses is that decision trees have a tendency to grow too large and thus overfit to the training data, i.e learning relationships within the data that is essentially noise, resulting in a significantly higher generalization error than training error. Another problem is instability and high variance, as small changes can lead to very different splits, and thus very different tree structures [Ruíz et al., 2015].

There are different ways to ensure the tree produced does not get too large. One of these methods is called *pruning*. Pruning is an approach where the pruning algorithm start at the bottom of the tree and walk through each node, replacing it with either one of its subtrees or a leaf. The replacement is decided based on which of the alternatives minimize the performance measure [Shalev-Shwartz and Ben-David, 2014]. This procedure will then hopefully return a tree with a reduced size.

4.4.2 Random forest

The random forest algorithm was introduced by Breiman [2001], and is essentially an ensemble of de-correlated decision trees. It is a further developed modification of *bagging*, which is is a method where you train many noisy and approximate models, and average their output in an effort to reduce variance [Ruíz et al., 2015]. In classification this ensemble is a committee of trees that vote for which class a given sample belongs to, with the final class being decided by the majority vote. In regression tasks, the prediction of the sample is an average of the numerical predictions from all ensemble members [Kuhn and Johnson, 2003].

This de-correlation of the trees are achieved by introducing randomness into the training. The randomness is introduced in two ways. One way is by training the individual trees on separate bootstrap datasets. This means we draw random samples from the original dataset with replacement [Ruíz et al., 2015]. In addition, for each node only a subset of the total number of features are evaluated for the split. For a dataset with k features, the subset of features evaluated for each split is often recommended to be $\sqrt{k}$, but this depends on the task

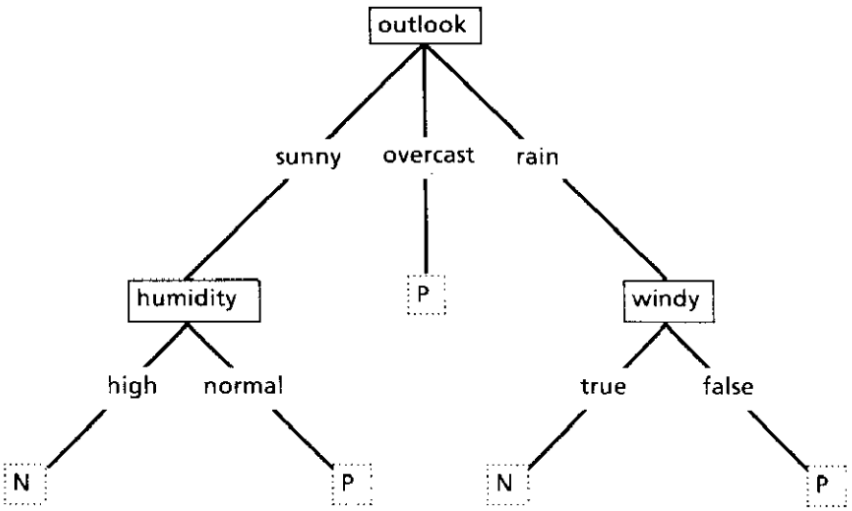


Figure 9: Schematic of a decision tree from Quinlan [1986]. Here *outlook* marks the *root node*, and the the three possible values of *outlook* (i.e *sunny, overcast, rain*) is branches from the root. While *overcast* lead directly to a *leaf node*, *sunny* and *rain* lead to the internal nodes of *humidity/windy*, which marks the root node of their own respective *subtree*.

at hand and is generally one of the hyperparameters that is tuned (more on this in sec. 4.5) [Kuhn and Johnson, 2003].

4.4.3 Neural network

Artificial neural network (ANN), from this point forward only referred to as neural network, is a learning algorithm which is based on the structure of the human brain, and tries to emulate logic reasoning seen in humans. Feedforward neural networks, also called multilayer perceptrons (MLP), is a type of neural network (NN) where information only flows forward through the model. In other words, the output of the model is not fed back into the model as part of the input for the subsequent step, i.e there is no feedback connections [Goodfellow et al., 2016]. Neural network which have these feedback connections are called recurrent neural network (RNN), and is often used to predict the next value in time-series, or in language detection (as the next letter in a sequence, i.e word, depends on the previous letters). As it is feedforward NN that is used in this thesis, RNN will not be explained further.

Consider an input $\mathbf{x}$ and output $\mathbf{y}$, where $\mathbf{y}$ can refer to either a class or a numerical value, and assume we can approximate a function f^* such that $y = f^*(x)$. A feedforward neural network would approximate f^* by defining

$$\mathbf{y} \approx f^*(\mathbf{x}; \theta) \tag{14}$$

where θ is parameters that needs to be learned in order to find the closest approximation of f^*.

A feedforward neural network is in practice structured by rather simple units referred to as units or *neurons*, which are densely interconnected in a network [Mitchell, 1997, Ruíz et al., 2015]. Each neuron takes an input and produces an output. Figure 10 shows a schematic of an (artificial) neuron, where $\mathbf{x}$ refers to the input to the neuron and $\mathbf{w}$ is the associated weights. $\mathbf{x}$ and $\mathbf{w}$ are used to calculate the weighted sum, which is then passed through the *activation function* $g(\cdot)$ to produce the neuron's output [Shalev-Shwartz and Ben-David, 2014]. The *activation function* is in principle a nonlinear transformation [Goodfellow et al., 2016], and many different functions can be used for this purpose. Two that are commonly used is rectified linear

units ($ReLU$) and *sigmoid* σ, defined in eq. 15 and 16 respectively.

$$g(\mathbf{x}) = ReLU = max(0, x) \tag{15}$$

$$g(\mathbf{x}) = \sigma(x) = \frac{1}{1 + exp(-x)} \tag{16}$$

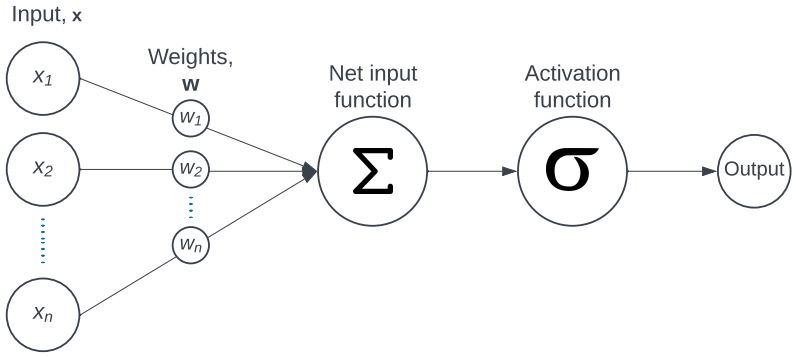


Figure 10: Schematic of an artificial neuron, based on Mitchell [1997] schematic of a perceptron. Each element of $\mathbf{x}$ is assigned a respective weight $\mathbf{w}$, and summed together. This weighted sum is then passed through the activation function, resulting in the neuron's output.

The neurons that make up the network are often structured in different *layers,* $V_0, .., V_T$, where the neurons in the tth layer V_t (where $t \in [T]$) work in parallel, while interconnected to neurons in the previous layer, V_{t-1} and the subsequent layer, V_{t+1}, as shown in fig. 11. How connected the neurons are between the layers are dependent on the type of neural network in question, and will not be discussed any further in this thesis. While θ is used to refer to the weights of the entire neural network, $\mathbf{w}$ is used to refer to the weights of a specific layer.

In order to get a better understanding of the concept of input and output of individual neurons and the connectivity between the layers, a slightly modified example from Shalev-Shwartz and Ben-David [2014] will be presented. Consider the ith of k neuron (where $i \in [k]$) within the tth layer V_t, i.e neuron $v_{t,i} \in V_t$. Let E refer to these interconnections between the neurons in V_t and the previous layer. $v_{t,i}$ input $a_{t,i}(\mathbf{x})$ is defined as

$$a_{t,i} = \sum_{r:(v_{t-1,r}, v_{t,i} \in E)} w((v_{t-1,r}, v_{t,i}))o_{t-1,r}(\mathbf{x}) \tag{17}$$

where $w((v_{t-1,r}, v_{t,i}))$ is the weight between neuron $v_{t-1,r}$ and $v_{t,i}$, and $o_{t-1,r}$ is the output from the rth neuron in the $(t-1)$th layer, V_{t-1}. In other words, the input to $v_{t,i}$ is essentially the weighted sum of the output of the neurons from layer V_{t-1} which has a connection to $v_{t,i}$. The output of $v_{t,i}$ is produced by passing input $a_{t,i}$ through the activation function g and defined as

$$o_{t,i}(\mathbf{x}) = g(a_{t,i}(\mathbf{x})) \tag{18}$$

Figure 11 illustrates this with three layers, V_0, V_1, V_3 which have $4, 5$ and 1 neurons in each respective layer. Consider the input and output of the first neuron in layer V_1, $v_{1,1}$, would from eq. 17 and 18 be

$$a_{1,1}(\mathbf{x}) = \sum_{r=1}^{4} w((v_{0,r}, v_{1,1}))o_{0,r}(\mathbf{x}), \quad \text{where } \mathbf{x} = [x_1, x_2, x_3, constant] \tag{19}$$

$$o_{1,1}(\mathbf{x}) = g(a_{1,1}(\mathbf{x})) \tag{20}$$

The bottom layer of the network is called the *input layer* and often has the dimension of $(p+1)$ for data X containing p samples, where the additional unit is called a *bias* unit, and often set to be 1 [Ruíz et al., 2015].
For a fully connected neural network, all neurons of one layer is connected to all neurons in the next, as seen in fig. 11. The diagram shows a feedforward neural network with one hidden layer. These layers can be expressed as nested functions, thus $f^*(\mathbf{x})$ can be expressed as

$$f^*(\mathbf{x}) = f^{(3)}(f^{(2)}(f^{(1)}(\mathbf{x}))) \tag{21}$$

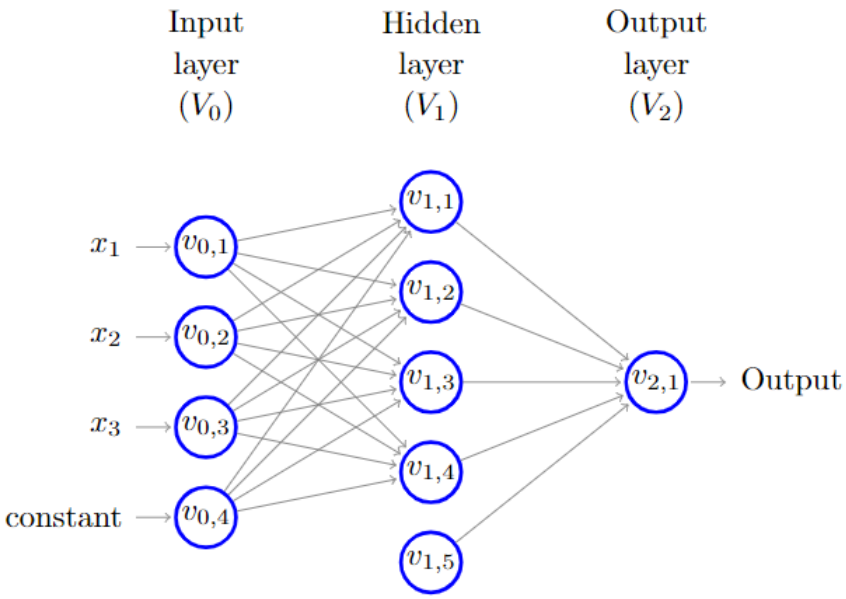


Figure 11: Schematic of a feedforward, one hidden layer neural network [Shalev-Shwartz and Ben-David, 2014]. The process showed in fig. 10 is happening within every individual neuron v. The output from each neuron in layer V_0 (input layer) provide the input to each individual neuron in layer V_1. For each neuron of layer V_1, the weighted sum of the input is passed through The activation function to create an output, which In turn will provide the output layer (V_2) with it's input.

The the different components making up the neural network, such as number of neurons and how these are organized in layers, as well as how each input and output unit is connected, is part of what is referred to as the *architecture* of the network [Goodfellow et al., 2016].

Training neural networks Back-propagation is a procedure introduced by Rumelhart et al. [1986] as "*repeatedly adjusting the weights of the connections in the network so as to minimize a measure of the difference between the actual output vector of the net and the desired output vector*". This is achieved by a two-step process; *forward pass* performs inference, and *backward pass* performs learning. In the forward pass, the input moves through each layer of the network with fixed weights until the network produces an output, $\hat{y}$, and the total error E can be calculated. The total error E can be defined in many ways, however, a measure that is commonly used is

$$E(\theta) \equiv \frac{1}{2} \sum_{d \in X} \sum_{k \in outputs} (\hat{y}_{d,k} - y_{d,k})^2 \tag{22}$$

where X is the training samples, $\hat{y}_{d,k}$ is the output produced by the neural network and $y_{d,k}$ is the target output in the supervised learning case [Mitchell, 1997]. The objective of back-propagation, specifically the backward pass, is to minimize E by adjusting the parameters θ, i.e

updates the weights such that the weight for the $(t+1)$ pass is updated according to

$$\theta_{t+1} = \theta_t - \gamma_t \nabla E(\theta_t) \tag{23}$$

where γ_t is the *learning rate*, a positive scalar determining the size of the step in the direction of the negative gradient, and $\nabla E(\theta_t)$ is the derivative of E [Goodfellow et al., 2016]. This update is done by gradient descent and requires the computation of the partial derivatives of E with respect to every weight $w_{t,i}$ (where $t \in [T]$ and $i \in [k]$) in the network [Rumelhart et al., 1986]. The vector derivative of E is thus defined by

$$\nabla E(\theta) = [\frac{\partial E}{\partial \theta_0}, ..., \frac{\partial E}{\partial \theta_n}] \tag{24}$$

where n is the total number of weights. Consider the gradient $\nabla E(\theta)$, which essentially is a vector in weight space. As it is a vector, it points to a specific direction, more specifically, the direction of the steepest increase in E [Mitchell, 1997]. Thus, in order to *minimize* E, we must move in the opposite direction of the gradient, hence the $-\nabla E(\theta_t)$ in eq. 23.

One downside to gradient descent is that it can be slow to reach the local minimum and computationally expensive as the number of training samples gets large, which is inconvenient as we know that the models ability to generalize well is very much dependent on large amounts of data. To account for this, one can compute noisy, approximate gradients instead of exact ones. This is called *stochastic gradient descent*. In contrast to general gradient descent that first update the weights after summing the error of all the training samples in X, stochastic gradient descent performs the back-propagation after every individual training sample, or alternatively sample a *minibatch* consisting of m training samples, and computing the gradient over these m samples in the minibatch [Mitchell, 1997, Goodfellow et al., 2016].
As the number of weights increase, the weight space can contain many local minimum. Because stochastic gradient descent takes steps in approximately the right direction, it may fail to fall into any of the local minimum. In addition, because of the often high dimensions of the weight space, the local minimum for one weight might not be the local minimum for another.

4.5 Hyperparameters in model selection

While some parameters are decided during the training process, others must be decide before the training even begins. These are called *hyperparameters* and are used to constrict the algorithm's behavior and learning. Examples of hyperparameters for random forest is maximum depth of the tree, number of estimators (i.e trees in the forest), learning rate and how many samples a node must contain to be a leaf. In neural networks there are not only hyperparameters that needs to be decided for each layer, such as activation function and learning rate, but also the number of hidden layers and the number of neurons within these layers, i.e the architecture of the network.
The hyperparameters can be optimized in a separate iterative algorithm before the training begins. This process is called *hyperparameter tuning*. Most learning algorithms have a vast number of possible hyperparameters, but usually only a select number of the hyperparameters are actually tuned (the remaining will have default settings), and these are selected based on the task at hand and the domain knowledge of the developer.
In order to evaluate which hyperparameters yield the optimized model, the performance of the different combinations of hyperparameter-settings need to be evaluated on different data than the training data. However, the evaluation of the final model requires previously *unseen* data, i.e the *test* data and this can therefor not be used in the hyperparameter tuning. Because of this, there is a need for another dataset, referred to as the *validation* data, which is specifically intended for the hyperparameter tuning.

Part III
Method

This part will describe the methods used in this thesis. Section 5 present the data, location and a brief description of the numerical model MEPS. Section 6 will present the methods used in feature selection, sec. 7 present the methods used in training and evaluation of the random forest classifier and sec. 8 the training and evaluation of the neural network.

5 Data

Model output (from the control run of the ensemble) from all four main cycles of MEPS has been selected for everyday of 2020 and 2021, in addition to January to August data for 2022, meaning we have four 66h-forecast for each date. In this thesis, model output for two main locations has been selected; Florida, Bergen (latitude: 60.38, longitude: 5.33) and Blindern, Oslo (latitude: 59.94, longitude: 10.72). Hanssen-Bauer and Førland [1998] defined 13 precipitation regions by using comparative trend analysis, and these two locations was chosen on because they belong to distinctly different precipitation regions. The defined precipitation regions was recently updated, and Bergen and Oslo belong to the updated regions of RR5 (Vestland, including the regions previously defined as RRX5 and RRX6) and RR2 (Østlandet, previously defined as RRX2) respectively [Hanssen-Bauer et al., 2022]. While the RR2 region has an average yearly precipitation of 941 mm (averaged over the period 1991-2020), the RR5 region has an averaged yearly precipitation of 2605 mm. The seasonal distribution of the precipitation also differ between the two regions. RR2 has on average the highest precipitation during summer (303 mm), while for the RR5 region winter has the highest precipitation (844 mm), followed closely by autumn (835 mm). For RR2, the spring is on average the season with the lowest precipitation of 167 mm. For RR5 on the other hand, the rank as least precipitating season is a tie between summer and spring, both with 479 mm.

Florida, Bergen is on the west coast of Norway, where the climate is characterized by mild winters and heavy precipitation due to orographic enhancement of frontal precipitation from low pressure systems. The climate at coastal areas have typically milder winters than areas further inland due to the heat capacity of the ocean, and this effect is even further intensified on Norway's west coast by the warm Gulf Stream, resulting in a yearly average temperature is $7.9°C$[Dannevig, a], with October having the highest monthly average precipitation, and May the lowest.

The general climate of Blindern, Oslo is characterized by southerly winds in the summer and northerly winds in the winter, resulting in relatively high summer temperatures, and cold winter temperatures, with the exceptions of areas surrounding Oslofjorden which experience relatively mild winters, and is also the area with the highest number of days with an average temperature above $20°C$. The yearly mean temperature is $6.2°C$ and the yearly mean precipitation for Oslo i particular is significantly lower than Bergen with $755mm$ [Dannevig, b].

These locations was specifically selected because of their difference in yearly average precipitation amount, as it was of interest to see how the machine learning algorithms performed on different precipitation distributions. Table 5 in sec. 5.3 will go further in detail about the difference in the percentage of the observations which is classified as precipitation between the two locations. All grid cells with an absolute distance of 3.5km from the main locations have been pulled, resulting in data from seven grid cells for both Florida and Blindern. The x, y pair

of each grid cell selected can be seen in tab. A.1 in App. A. In order to reduce noise, the data for each location was averaged across these seven grid cells.

5.1 MEPS

The Meteorological Cooperation on Operational Numerical Weather Prediction (MetCoOp) model is a convective-scale model operated by the national meteorological institutions of Norway, Sweden and Finland in a bilateral effort. The model MetCoOp Ensemble Prediction System (MEPS) produces a 66h forecast every 6 hours with nine perturbed members, at 00, 06, 12 and 18 UTC (referred to as the *main* cycles). In between these main cycles, the model is updated every 3h to produce a short-term forecast which is then used as a background field in the upcoming main cycle. MEPS has a horizontal resolution of $2.5km \times 2.5km$. As seen in fig. 12, the model cover most of the Nordic countries, as well as part of the Atlantic ocean, the Nordic sea and the Baltic sea [Müller et al., 2017].
MEPS is forced by ECMWF-IFS (European Centre for Medium-range Weather Forecasts Integrated Forecasting System) at the lateral and upper boundaries, and is based on the HARMONIE-AROME configuration of the ALADIN-HIRLAM system [Frogner et al., 2019].

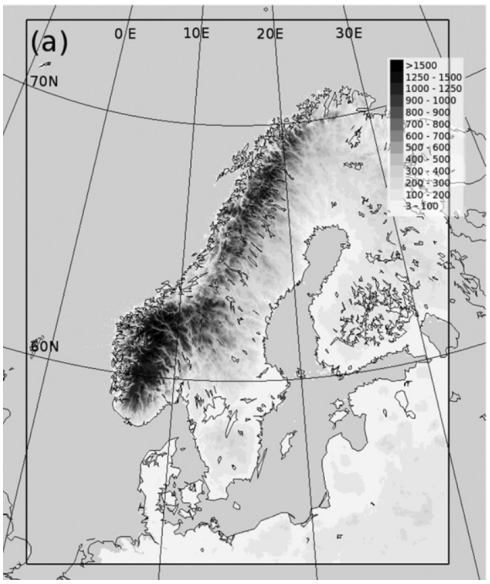


Figure 12: Contoured land topography for the entire model domain of MEPS. [Müller et al., 2017]

One thing to note, is that there is an update in both model output and and filename after 4 February 2020 06 UTC. Prior to this point, the model data is stored as $MEPS\_subset\_2\_5km$ and includes the *air temperature at lowest model level* variable. After the update at 4 February 2020, the model output is stored as $MEPS\_det\_2\_5km$, and no longer explicitly includes the temperature variable mentioned previously. The *subset* file contained the output from all ensemble members in addition to the control run, while the *det* file only contain output from the control run. In addition, a few new variables has been added, as seen in tab. 1.

Table 1: This table gives an overview of changes in the variables included after the MEPS update on 4 April 2020. Only changes of variables used in this thesis is included.

New variables included after update	**Unit**
Convective Available Potential Energy (CAPE)	Jkg^{-1}
Atmosphere convective inhibition (CIN)	Jkg^{-1}
TOA SW net clear sky radiation	$Wm\ -2$
Vertical wind pressure levels	ms^{-1}
Old variables not included after update	**Unit**
Air temperature at lowest model level	K

To accommodate for this change, the *air temperature at lowest model level* is pulled from the lowest model level of the variable *air temperature model level* from 4 February 2020 12UTC and beyond. The variables available after the update, has all been given NaN values for the time period prior to 4 February 2020 12 UTC. For unknown reasons *atmosphere convective inhibition (CIN)* is missing from the model output in a period expanding from 24 March 2021 12 UTC until 20 May 12 UTC. This results in a slight increase of missing values in the dataset.

5.2 Datasets

In this thesis the machine learning algorithms have been trained on different datasets for comparison. Two different locations (Florida and Blindern), as well as two sets of features (referred to as 01 and 02) included in the dataset and a variety of forecast length has been tested. An explanation of the abbreviation used to reference these differences can be seen in the upper panel of table 2. The lower panel shows the full list of the different names of all datasets, and the datasets will be referenced by these names in the rest of the thesis.

Table 2: (upper panel) An overview and description of the abbreviations used for the different locations, dataset features and forecast lenght. (Lower panel) An overview of the name of each dataset.

	Abbriviation	**Definition**
Location	F, B	Florida, Blindern
Set of variables	Dataset 01	Features from MEPS only
	Dataset 02	01 features + additional features calculated from the MEPS variables
Forecast length	P01	Full forecast lead time $[0, 66]$ hours)
	P02	Dataset split into three subset depending on forecast lead time. The three subsets is 0-6 hours, 6-12 hours and 12-66 hours. Referenced as: P02-6h, P02-12h and P02-66h
Dataset overview	F01P01 / B01P01	MEPS variables, forecast lead time: [0,66]h
	F01P02 / B01P02	MEPS variables, forecast lead time: [0,6]h, [6,12]h,[12,66]h
	F02P01 / B02P01	MEPS with additional calculated features, forecast lead time: 66h
	F02P02 / B02P02	MEPS with additional calculated features, forecast lead time: [0,6]h, [6,12]h,[12,66]h

One thing to note is that the P02 datasets (i.e F01P02, B01P02, F02P02, B02P02) is in practice three datasets each, and is thus referenced as P02-6h, P02-12h and P02-66h, corresponding to the forecast lead time of $[0, 6]h, [6, 12]h$ and $[12, 66]h$ respectively.

Table 3 shows an extensive overview over all variables included in dataset 01, while tab. 4 shows an overview of the additional variables included in dataset 02. A a reminder, dataset 02 is a combination of *both* sets of variables found in tab. 3 and 4.

F01/B01: MEPS

Table 3: Variables directly retrieved from MEPS

Variable	**Unit**
Air temperature at 2m	K
Accumulated total precipitation	kgm^{-2}
Total accumulated solid precipitation (snow+graupel+hail)	kgm^{-2}
Mean Sea Level Pressure (MSLP)	Pa
Screen level temperature	K
Screen level relative humidity	1
Air temperature at lowest model level	K
Zonal 10 meter wind	ms^{-1}
Meridional 10 meter wind	ms^{-1}
Accumulated surface SW downwelling radiation	Wsm^{-1}
Height of PBL	m
Cloud cover of high clouds (HCC)	1
Cloud cover of low clouds (LCC)	1
Cloud cover of medium height clouds (MCC)	1
Total cloud cover (TCC)	1
Fog	1
Air temperature at pressure level 925hPa	K
Air temperature at pressure level 850hPa	K
Air temperature pressure levels	K
Geopotential at pressure level 500 hPa	m^2s^{-2}
Relative humidity at pressure level 925hPa	1
Relative humidity at pressure level 500hPa	1
Zonal wind at pressure level 925hPa	ms^{-1}
Zonal wind at pressure level 850hPa	ms^{-1}
Zonal wind at pressure level 500hPa	ms^{-1}
Meridional wind at pressure level 925hPa	ms^{-1}
Meridional wind at pressure level 850hPa	ms^{-1}
Meridional wind at pressure level 500hPa	ms^{-1}
Vertical wind at pressure level 925hPa	ms^{-1}
Vertical wind at pressure level 850hPa	ms^{-1}
Vertical wind at pressure level 500hPa	ms^{-1}
Convective Available Potential Energy (CAPE)	Jkg^{-1}
Atmosphere convective inhibition (CIN)	Jkg^{-1}
TOA SW net clear sky radiation	$Wm\ -2$

F02/B02: Additional calculated variables

Table 4: Additional variables calculated from MEPS variables. Note that the variables *relative humidity at 850/700hPa* and *air temperature at 700hPa* and all of *specific humidity* is direct MEPS output not part of dataset 01, but was needed to calculate some of the variables in 02.

Variable	Unit
MEPS: Label, y	1
Relative humidity at 850hPa	1
Relative humidity 700hPa	1
Air temperature 700hPa	K
Dewpoint temperature at 925hPa	$°C$
Dewpoint temperature at 500hPa	$°C$
Dewpoint temperature at 2m	$°C$
Specific humidity at 925hPa	$\frac{kg}{kg}$
Specific humidity at 500hPa	$\frac{kg}{kg}$
Windspeed at 10m	ms^{-1}
Windspeed at 925hPa	ms^{-1}
Windspeed at 850hPa	ms^{-1}
Windspeed at 500hPa	ms^{-1}
Wind direction at 10m	°
Wind direction at 925	°
Wind direction at 850	°
Wind direction at 500	°
Pressure of lifting condensation level	Pa
Temperature of lifting condensation level	$°C$
Absolute moisture flux at 925hPa	$\frac{kg\ m}{kg\ s}$
Zonal moisture flux at 925hPa	$\frac{kg\ m}{kg\ s}$
Meridional moisture flux at 925hPa	$\frac{kg\ m}{kg\ s}$
Specific humidity at 2m	$\frac{kg}{kg}$
Windspeed tendency 10m	$\frac{ms^{-1}}{hour}$
Windspeed tendency 925hPa	$\frac{ms^{-1}}{hour}$
Windspeed tendency 850hPa	$\frac{ms^{-1}}{hour}$
Windspeed tendency 500hPa	$\frac{ms^{-1}}{hour}$
Air temperature tendency 2m [dT/dt]	$\frac{K}{hour}$
Relative humidity tendency 2m [dRH/dt]	$\frac{1}{hour}$
Zonal wind tendency 10m [du/dt]	$\frac{ms^{-1}}{hour}$
Meridional wind tendency 10m [dv/dt]	$\frac{ms^{-1}}{hour}$
Pressure tendency [dp/dt]	$\frac{Pa}{hour}$
Zonal wind tendency at 925hPa [du/dt]	$\frac{ms^{-1}}{hour}$
Zonal wind tendency at 850hPa [du/dt]	$\frac{ms^{-1}}{hour}$
Zonal wind tendency at 500hPa [du/dt]	$\frac{ms^{-1}}{hour}$
Meridional wind tendency at 925hPa [dv/dt]	$\frac{ms^{-1}}{hour}$
Meridional wind tendency at 850hPa [dv/dt]	$\frac{ms^{-1}}{hour}$
Meridional wind tendency at 500hPa [dv/dt]	$\frac{ms^{-1}}{hour}$
Air temperature tendency at 925hPa [dT/dt]	$\frac{K}{hour}$
Air temperature tendency at 850hPa [dT/dt]	$\frac{K}{hour}$
Air temperature tendency at 500hPa [dT/dt]	$\frac{K}{hour}$
Relative humidity tendency at 925hPa [dRH/dt]	$\frac{1}{hour}$

Continued on next page

Table 4: Additional variables calculated from MEPS variables. Note that the variables *relative humidity at 850/700hPa* and *air temperature at 700hPa* and all of *specific humidity* is direct MEPS output not part of dataset 01, but was needed to calculate some of the variables in 02. (Continued)

Variable	Unit
Relative humidity tendency at 500hPa [dRH/dt]	$\frac{1}{hour}$
Dewpoint temperature at 700hPa	$°C$
Dewpoint temperature at 850hPa	$°C$
K index	$°C$
Precipitable water	mm

5.3 Observations

In addition to the numerical model data, one must also provide the machine learning algorithms the corresponding target of the training data (in supervised learning). In this thesis, the classes in the classification task are defined as 0 : *no precipitation* and 1 : *precipitation*. Hourly accumulated precipitation observations from each location have been used to decide which class the training data will be assigned, as well as provide the target value of the regression task. This observational data has been downloaded from Klimaservicesenter. At each location there are several stations which could provide the observational data needed. Table 5 shows which stations the observational data has been pulled from, as well as the percentage of the total number of samples which classify as precipitation. These stations were chosen because they contain no missing data in the period 2020-2022. If the hourly accumulated precipitation measured $\geq 0.1 mmh^{-1}$ it was classified as 1 : *precipitation*, and as 0 : *no precipitation* otherwise.

Table 5: An overview of the stations used to find observations for both locations for the year 2020, 2021 and 2022. An overview of number of observations, number and percentage of observations classified as class 0 (*no precipitation*) and class 1 (*precipitation*).

	Station name	# of obs	Precipitatin	No precipitation
Blindern	SN18700	22 754	12.7%	86.0%
Florida	SN50540	22 754	28.7%	71.3%

5.4 Handling missing data

As mentioned in section 5.1, there is a slight change in variables included in the MEPS output after 4 February 2020. Because of this, the variables shown in table 1 contributes to a greater percentage of NaN values within the dataset before 4 February 2020, compared to after. The random forest classifier (RFC) from the Python package scikit-learn is used in the first machine learning step (described further in sec. 7), and the algorithm is incompatible with NaN values [Buitinck et al., 2013]. In order to accommodate for this all rows containing NaN values was removed from the dataset. This reduced the dataset for Florida from 252 724 samples to 217 006 samples, and from 252 724 to 224 266 for Blindern. Although this results in a slightly smaller dataset, the resulting dataset still contains a sufficient amount of samples.

6 Feature engineering

This thesis looks at two different sets of features. Dataset 01 (presented in tab. 3), which contains features pulled directly from MEPS [The Norwegian Meteorological Institute (MET Norway), 2022] (with the exception of the calculated variable *hourly precipitation*, which is calculated using $hourly\_precip = X(t) - X(t-1)$, where X is *total accumulated precipitation amount* from MEPS). Dataset 02 contains all the same variables as 01, but additional 50 variables (presented in tb. 4) was calculated using the original variables, resulting in a total of 80

features in Dataset 02. The overview of how the additional features was calculated is found in App. A.2.

The data was split into *training*, *validation* and *test* data, with the training containing 60% of the full dataset, and the remaining 40% split equally between the validation and test data. Although it would be more ideal to split the data chronologically due to the auto-correlation within the data [Schultz et al., 2021], the samples for the different subsets was originally pulled at random. The reason for this was the limited data period. When splitting the data chronologically, the validation data was pulled exclusively from the summer months of 2021 (model data was originally only from 2020-2021), meaning the test data was not representative as it was a significant decrease in precipitation occurrence compared to the training set. Doing a random split ensured an equal distribution of precipitation occurrence between the different datasets. In order to split chronologically, one should preferably have many years of data to make sure the training, validation and test set are all representative of the population it is drawn from. Because of the limited data period a chronological split did not satisfy this, thus a random split was favored. However, upon further inspection of the performance of the machine learning algorithms on unseen 2022 data (as an extra step after model evaluation using the randomly drawn test data) it was deemed necessary to redo the process with chronologically split data. The reason for this will be further presented and discussed in sec. IV.

6.1 Feature selection

As mentioned in section 4, one of the most important aspects of training a machine learning (ML) algorithm is to do proper feature selection/engineering. This section will go through the steps taken in the selection process.

6.1.1 Exploratory data analysis

Exploratory data analysis was done using the methods described in section 3. This includes calculating the Pearson correlation and Spearman rank between the $i$the feature and the observed precipitation measured at the location in question, as well as the correlation matrix (using Spearman rank, thus will refer to as the rank matrix from this point forward). Using the rank matrix, a Dendrogram was then plotted to visualize the hierarchical clustering of the features.

6.1.2 Base model and feature importance

Sklearn's RandomForestClassifier (RFC) was the random forest algorithm used to classify the samples into *no precipitation* and *precipitation*. The forest is built from many individual trees trained on a bootstrap sample drawn from the training data, and for each node, only a subset of the total number of features are searched to find the best split [Scikit-learn User-Guide, 2022, chap. 1.11.2]. These two properties makes the forest more randomized and diverse, reducing overfitting. The final prediction from the forest is given by the averaged predictions from each tree. A small forest with strict, shallow trees was initiated with the hyperparameters shown in table 6. This estimator will from this point forward be referred to as the *base* RFC. The *base* RFC was then fitted on all features in the given dataset. *Sklearn's* random forest algorithm computes a property called *feature importance*, and Scikit-learn API Reference [2022] states that this is the the total reduction of the chosen criterion brought by the particular feature split. In other words, the higher the feature importance, the higher the gain measure is for the particular feature. The feature importance is a value between 1 and 0, and the sum off the feature importance of all features is 1. In practice this means that the feature importance's of dataset 02 (82 features) will have lower values than dataset 01 (33 features).

Table 6: Table shows all hyperparameters used to initiate the *base* RFC. The following were the only hyperparameters specified, the remaining settings were set to their default value (for full overview [see Scikit-learn API Reference, 2022, module: Random Forest Classifier])

bootstrap	max features	max samples	min sample leaf	min sample split	oob score	random state	criterion	max depth	n estimators
True	'auto'	0.5	150	500	True	0	'gini'	5	50

6.1.3 Recursive feature elimination

Another approach to feature selection is to do *recursive feature elimination*. In addition to the steps described above, *sklearn's* recursive feature elimination with cross-validation (RFECV) was used. The RFECV algorithm performs recursive feature elimination by fitting the estimator to the full set of features, looking at the feature importance's and then pruning the least important features from the dataset. This process is repeated on the pruned set, and this is recursively done until the optimal number of features has been selected. The algorithm uses cross-validation in order to decide the optimal number of features (from this point referred to as n [Scikit-learn User-Guide, 2022, chap. 1.13.3]. From this, the objective used by the cross-validation as a function of n was plotted to check whether the selected n was the most sensible choice, or if the objective plateaued and achieved a sufficient performance at a lower n. In instances where this plot indicated a lower n might be sufficient, *sklearn's* recursive feature elimination (RFE) algorithm was initiated. Instead of using cross-validation to find the optimal n, RFE takes n as an input from the user, and repeats the process of recursive feature elimination in the same manner as the RFECV algorithm, until (the user-specified) n features have been selected. The n specified in the hyperparameter-settings of the RFE algorithm was found by a visual inspection of the objective versus n plot (from the RFECV), as well as a comparison of the n found for similar datasets.

6.1.4 Final feature selection

The last step of the feature selection process compares the verification metrics of the *base* RFC (the base random forest classifier trained on *all* features), RFECV (*base* RFC trained on the RFECV-selected features), and in the cases where relevant, the RFE (*base* RFC trained on the RFE-selected features). The features selected by RFECV/RFE are also compared to the Spearman rank and feature importance as a way of checking the validity of the algorithm's choices.

7 Random forest classifier

This section will go through every step of the process of training a random forest classifier (RFC), from model selection to final model evaluation. This process is done on every individual dataset.

7.1 Model selection and evaluation

7.1.1 Model selection

Deciding which hyperparameters to use for the machine learning (ML) algorithm of choice is an important step. As mentioned in section 4, the ML algorithm will display different behavior based on the choice of hyperparameters, and by extension this means the hyperparameters also influence performance of the final ML algorithm. The process of finding the optimal hyperparameters for the given task is also referred to as model selection. In order to optimize

the hyperparameters, *sklearn's* GridSearchCV has been used, which is a algorithm used for hyperparameter tuning. The user decide which hyperparameters to search and different discrete settings to try for each of the parameters of choice. This set of hyperparameters and their discrete settings are referred to as the *hyperparameter space*. The GridSearchCV works by taking a given algorithm and the hyperparameter space as input. It will search the hyperparameter space to find the best combination by doing an exhaustive calculation of the objective of each combination, and return the best performing model [Buitinck et al., 2013]. In order to calculate the objective, the algorithm was set to use the cross-validation scheme, *5*-fold. Though the objective is by default set to be the same as for the algorithm used as input, the user has the option to specify a objective of their own choosing. The objective used in this thesis is *balanced accuracy* and is defined as [Scikit-learn API Reference, 2022, chap. 3.3.2.4]:

$$Balanced\,accuracy = \frac{\frac{a}{a+c} + \frac{d}{b+d}}{2} \tag{25}$$

where a, b, c, d follow the definition in tab. 9 in sec. 2.6 The reason this objective was selected was because of the unbalance of the classes. Traditional scoring like accuracy $((a+d)/(a+b+c+d))$ is not the best choice when the data is unbalanced, as the model can perform very well by only predicting the majority class. While *balanced accuracy* was used for all random forest classifiers, a trial was done using the *F-measure*, F_1, on the F01P01 dataset to see whether this particular scoring would provide a better hyperparameter search. Better is here meant to mean more aligned with the verification measured traditionally used in forecast verification, which will be described later in sec. 8.3. Why this was deemed necessary will become clear in sec. 12.1). F_1 is defined as [Scikit-learn API Reference, 2022, chap. 3.3.2.9]:

$$F_1 = \frac{2 * \frac{a}{a+c}\frac{a}{a+b}}{\frac{a}{a+c} + \frac{a}{a+b}} \tag{26}$$

The hyperparameters used in the initiation of the grid search is shown in table 7, where the gray highlighted marks the hyperparameter space, while the other parameters stay fixed for each dataset.

7.1.2 Model evaluation

The final step in the process is to evaluate the random forest classifier (RFC) on unseen data (*i.e test data*) using the metrics for deterministic, discrete forecast presented in section 8.3.

Table 7: Table shows all hyperparameters used in the model selection. Describtion is reproduced from [Scikit-learn API Reference, 2022, module: Random Forest Classifier].

Hyperparameters	Setting	Description
bootstrap	True	Bootstrap sampling is used when building the individual trees
max_features	'auto'	Maximum number of features to search at each split. 'auto' corresponds to $\sqrt{n\ features}$
max_samples	0.5	(Since *bootstrap=True*) The number of samples to draw from the dataset for each tree.
min_sample_leaf	150	$min(n\ samples)$ required to be a leaf node
min_sample_split	500	$min(n\ samples)$ to split internal node
oob_score	True	Use out-of-bag samples to estimate generalization score.
random_state	0	Set to control the randomness of drawing sampling for Bootstrap (for reproducibility)

Continued on next page

Table 7: Table shows all hyperparameters used in the model selection. Describtion is reproduced from [Scikit-learn API Reference, 2022, module: Random Forest Classifier]. (Continued)

Hyperparameters	Setting	Description
ccp_alpha	[0.0, 0.01, 0.015, 0.02]	Pruning coefficient, where 0.0 means no-pruning. For further explination, [see Scikit-learn API Reference, 2022, module: Random Forest Classifier]
criterion	['entropy', 'gini']	The criterion used to decide the split. To view formulas, see section 4.4.1
class_weight	[None, 'balanced', 'balanced_subsample']	None: all weights are set to 1. 'balanced': y is used to calculate the frequency of each class. The weights are then adjusted to be inverse of the frequency. 'balanced_subsample': same as 'balanced', but the freqency is found from the Bootstrap resample group, not the full dataset.
max_depth	[5,10,25,50]	Maximum depth of each tree
n_estimators	[50, 250, 500, 750]	Number of trees the forest should consist of.

8 Neural network

After using the random forest classifier (RFC) to classify the data into *no precipitation* and *precipitation*, a separate neural network (NN) was trained for each of the classes to make a deterministic forecast of the *hourly accumulated precipitation amount*. Because of the results of the work described in sec. 7, it was decided to only train the neural networks on P01 datasets, i.e the full forecast period.

8.1 Base NN

A base neural network was built containing *one* hidden layer using Tensorflow's *Keras* package, based on the Sequential model class. This basic neural network will be referred to as the *base* NN from this point forward. The Sequential model is, just as the name attributes, a model built by stacks of layers. After initiating the model, one can add the subsequential layers wanted, with the type of layer dependent on what type of neural network one intends to build. In order to build a basic neural network, the Keras *Dense* layer, which is a densely-connected layer [TensorFlow, 2015], was used to build both the *base* NN and the optimized neural network described in the following sub-section.
The architecture for the *base* NN consist of an input layer with 64 neurons, a hidden layer with 64 neurons, and an output layer with *one* neuron (due to it being a regression task). The activation function for all layers was set to ReLu, which is defined in eq. 15 in sec. 4.4.3.

When fitting the *base* NN to the training data, the mean absolute error (MAE as described later in sec. 8.3) was used to assess the loss, and the number of epochs was set to 100. Initially, the same features selected for the random forest classifier (RFC) was used to train the *base* NN. Because we have auto-correlated variables and the data structure not favorable for recurrent neural networks, the tendency features calculated for the 02 dataset was introduced back into the dataset for the *base* NN in an effort to encapsulate some of the temporal dependencies. An overview of the additional features used in the neural network training is highlighted (gray) in tab. 4.

8.2 Model selection

The *base* neural network (NN) was trained with the features selected in the recursive feature elimination,both with and without the tendency variables. After assessing the performance and whether to include the tendency features in the further training, a model selection was done using *KerasTuner* from O'Malley et al. [2019]. KerasTuner *RandomSearch* algorithm

was the search-engine used, which in each trial builds a model based on a randomly-selected hyperparameter-combination from the search space, fits the model to the training data, before evaluating the model on the validation data. After all the trails have been executed, the best model is selected based on the objective specified by the user (here *validation MAE*), and the optimal hyperparameters found from this. Because RandomSearch test a finite number of hyperparameter-combinations (*max_trails* argument, here set to 10), and not all possible combinations, it is not necessarily the best global combination of hyperparameters that is selected, but a local.
The hyperparameters that was optimized includes *number of hidden layers* as well as the *number of neurons* of each layer, and table 8 shows the range of values being search. While the activation function for the input layer and all hidden layers was set to $ReLU$, the output layer had the option between $ReLU$ and *sigmoid*, the latter defined in eq. 16 (sec. 4.4.3).

Table 8: The hyperparameters making up the search space. The upper panel shows the hyperparameters related to the architecture of the network, i.e number of hidden layers and number of neurons making up each layer. The lower panel shows the hyperparameter related to the output-layer specificly.

Hyper-parameter	Setting	Description
	Architecture of the network	
num_layers	$min = 2, max = 12]$	Number of layers. The range starts at 2 in order to get a minimum of one hidden layer, as the first layer is the input layer.
units	$min = 32, max = 128]$, $step = 32$	Number of neurons in each layer.
	Output layer	
activation	['ReLU', 'sigmoid']	Activation function

After the optimized structure and hyperparameters of the neural network was selected, the neural network was retrained with both training and validation data, before the neural network was evaluated using the unseen *test* data. Due to the results showing that the 01 datasets, both with and without the tendency features, performed slightly better than the 02 dataset, further forecast verification was only performed on the 01-based forecast.

8.3 Forecast evaluation

8.3.1 Deterministic forecast of discrete variables

In this thesis the classification of *no precipitation/precipitation* provided by the random forest classifier (RFC) is equivalent to a deterministic forecast of a discrete variable, where the discrete variable is hourly accumulated precipitation $\geq 0.1\, mmh^{-1}$. In order to evaluate this forecast, certain scores must be calculated, as illustrated by the contingency table in tab. 9. *True positives*, referred to as a in the contingency table, gives a measure of how many of the predicted *yes*-events actually manifested, while *false positives (d)* measures how many of the predicted *yes*-events failed to manifest. In contrast, *false negatives (c)* measures how many of the observed *yes*-events the forecast failed to predict, and *true negative (d)* measures how many predicted *no*-event occurred.

Table 9: The contigency table show the four different outcomes of a forecast of discrete variables. Outcome a refers to a *True positive*, outcome b refers to a *False positive*, outcome c refers to *False negative*, while outcome d refers to a *True negative*.

		Observed		
		Yes	**No**	Total
Forecast	**Yes**	a	b	$a+b$
	No	c	d	$c+d$
	Total	$a+c$	$b+d$	$n=a+b+c+d$

The different measures provided in the contingency table can be used to calculate many different verification metrics. The sum of *true positives* and *true negatives* divided by the total number of events, n, result in the accuracy (ACC) measure (also referred to as proportion correct, PC) defined as

$$ACC = \frac{a+d}{n} \tag{27}$$

The number of *true positives* divided by the total number of observed *yes*-events provide a metric measuring the fraction of observed *yes*-events that is accurately predicted by the forecast, and is referred to as the *probability of detection (POD)* or *hit rate, H*

$$POD = \frac{a}{a+c} \tag{28}$$

The counterpart of POD is namely the *probability of false detection (POFD)*, and is a measure of the fraction of observed *no*-events which are predicted as *yes*-events (i.e *false positives*), and defined by

$$POFD = \frac{b}{b+d} \tag{29}$$

and can also be referred to as the *false alarm rate*. This should not be confused with the verification metrics termed similarly as the *false alarm ratio (FAR)*, which is a measure of the fraction of predicted *yes*-events that fail to manifest

$$FAR = \frac{b}{a+b} \tag{30}$$

POD attributes equal weight to *true positives* and *true negatives*, and in instances where one event occurs much more often than the other, which is the case of *no precipitation* and *precipitation*, this unbalance is lost. When the *no*-events occur much more frequently than the *yes*-events, the *critical success index (CSI)* or *threat score, TS*, defined as

$$CSI = \frac{a}{a+b+c} \tag{31}$$

might be a better measure. sucess ratio (SR) is defined as

$$SR = \frac{a}{a+b} \tag{32}$$

(Multiplicative) Bias is defined as the total number of predicted *yes*-events over the total number of observed *yes*-events,

$$B = \frac{a+b}{a+c} \tag{33}$$

where $B = 1$ is an unbiased forecast, $B \leq 1$ means the forecast under-predicts the occurrence, and $B \geq 1$ means the forecast over-predict the occurrence.

In order to compare the performance of the machine learning algorithms with the performance of MEPS, the *skill score (SS)* defined as

$$SS_{ref} = \frac{ACC - ACC_{ref}}{ACC_{perf} - ACC_{ref}} \tag{34}$$

is calculated, where ACC_{ref} is the accuracy achieved by guessing the majority class (*no precipitation*) for every sample, and ACC_{perf} is the accuracy achieved by a perfect forecast, i.e 1.

8.3.2 Deterministic forecast of continuous variables

When evaluating the quality of a forecast of continuous variables, a metric commonly used to measure accuracy is the *mean absolute error (MAE)*, which gives the averaged absolute difference between the forecast and observations, defined by

$$MAE = \frac{1}{n}\sum_{k=1}^{n}|y_k - o_k| \tag{35}$$

where the forecast-observation pair (y_k, o_k), is the kth of a total of n pairs [Warner, 2011].

While the MAE gives the absolute difference, the *mean-squared error* provide the average squared difference, and is defined as

$$MSE = \frac{1}{n}\sum_{k=1}^{n}(y_k - o_k)^2 \tag{36}$$

Because the forecast error is squared, large errors will give have an greater impact on the MSE compared to the MAE.

The average difference between the forecast and the observation is termed the *Mean error* and is defined as

$$ME = Bias = \overline{y} - \overline{0} \tag{37}$$

This is also called (additive) bias, where $bias = 0$ refers to an *unbiased* forecast. Note that this is in contrast to the (multiplicative) bias defined for discrete predictands (which is *unbiased* when $bias = 1$).

The *skill score* for continuous variables is

$$SS_{ref} = 1 - \frac{MAE}{MAE_{ref}} \tag{38}$$

where MAE_{ref} is the MAE of a reference forecast, in this thesis a reference forecast of only guessing $0\,mmh^{-1}$.

8.3.3 Probabilistic forecast of discrete variables

In order to look further into the capabilities of the random forest classifier (RFC), the RFC was set to predict the class probabilities. The probability of precipitation was then plotted in a Relative Operating Characteristic (ROC) curve and a reliability diagram as described below.

ROC curve The Relative Operating Characteristic (ROC) curve is a way to graphically display a forecast of discrete variables, with *probability of false detection* on the x-axis and *probability of detection* on the y-axis [Wilks, 2011]. The plot shows these two verification metrics for different threshold probabilities. The threshold probability determines the required probability to forecast *precipitation*, and is varied in the rang of $[0, 1]$, where a threshold of 0 corresponds to always forecasting *precipitation*, and 1 corresponds to always forecasting *no precipitation*.

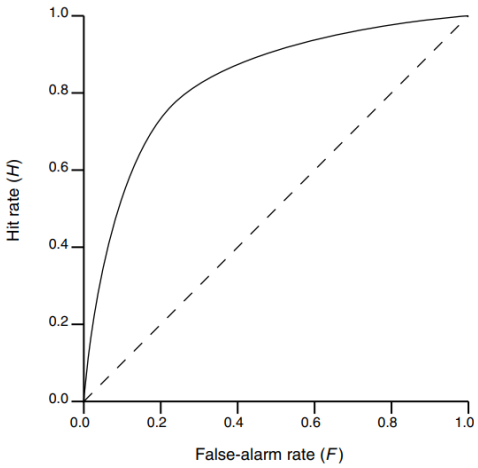
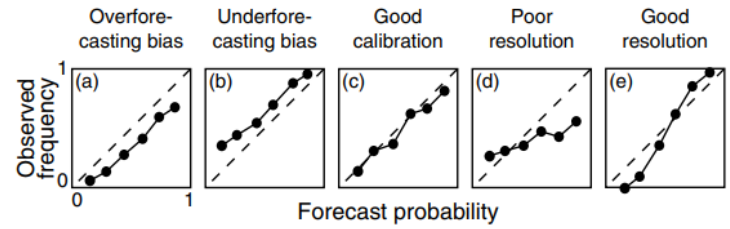
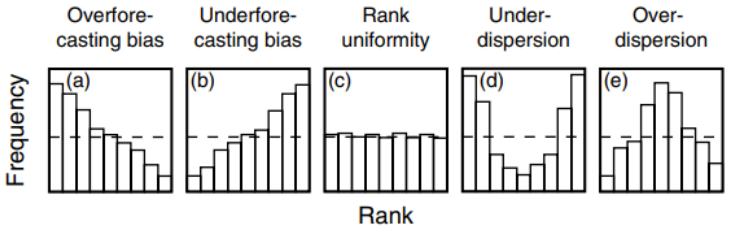


Figure 13: Figure 14a shows a schematic of the ROC curve, with the POFD (probability of false detection) on the x-axis (referred to as the False alarm rate F) and POD (probability of detection) at the y-axis (referred to as Hit Rate H). The plot shows the POFD and POD for different threshold probabilities of precipitation. Figure from Warner [2011]

Reliability diagram The reliability diagram consist of two parts, the reliability graph, shown in fig. 14a, and a rank histogram as shown in fig. 14b. The reliability graphs have forecast probability at the x-axis and observed frequency at the y-axis, meaning a plot centered around the one-to-one line means the forecast has good calibration. A graph positioned below this line means there is an overprediction bias, while on the other hand a graph above the line shows a underprediction bias. The rank histogram, with forecast probability on the x-axis and frequency at the y-axis, show if there is a systematic bias in the forecast. Plot *(c)* in fig. 14b show the wanted result of rank uniformity.



(a)



(b)

Figure 14: Figure 14a shows a schematic of a reliability graph, where the x-axis shows the forecast probability and y-axis show observed probability. A graph centered at the one-to-one line refers to a forecast with good calibration (c), while (a) and (b) show a systematic over- and underforecasting bias respectively. Fig. 14a. Figures from Warner [2011].

Part IV
Results and discussion

Due to the fact that this thesis has three distinct and subsequent steps it is necessary to discuss and conclude each step before presenting the results of the next. The structure of the following part will thus be as follows; the results of the particular step in question will be presented, discussed and concluded before moving onto the next one. As described in the method in sec. III, the steps are in the following order; feature selection, classification of *no precipitation/precipitation* by random forest classifier (RFC, see sec. 7) and lastly predicting *hourly accumulated precipitation amount* by neural networks (NN, see sec. 8).

9 Different strategies for splitting data

As mentioned in sec. 6 it was decided to split the data randomly in order to get an even distribution of precipitation occurrences across the training, validation and test data. Feature selection, model selection and model evaluation described in section III was executed using these randomly split datasets in order to train a random forest classifier (RFC) and a neural network (NN). In an effort to see how well the 2-step machine learning system perform on entirely new data, it was used to make a *hourly accumulated precipitation amount* forecast using MEPS data (for Florida) from January-August 2022. The results for this was rather inconsistent with the model evaluation done using the randomly drawn test data (i.e unseen data not used in the training of the model), and had an significantly poorer performance. See tab. D.16 and D.17 in App. D.1 to see the full comparison in mean absolute error. The evaluation done on randomly drawn test data showed a 20% *reduction* in mean absolute error (MAE) for class 0 (*no precipitation*) for the NN compared to MEPS, while this same NN evaluated on 2022 data showed a 2.7% *increase* in MAE for this class. An even greater difference is seen for class 1 (*precipitation*) where the NN got a *reduction* in MAE of 30% in comparison with MEPS when evaluated on randomly drawn test data. For the 2022 data however, the NN *increased* the class 1 MAE with 8.3% compared to MEPS. This illustrates just how optimistic the results can be when evaluated on randomly drawn data, just as emphasized by Schultz et al. [2021]. The auto-correlation within the data cause cross-contamination across the training, validation and test data, giving overly optimistic results and thus make the final model performance poorer on entirely new data.

Because of this clear inconsistency of the model performance on the 2022 data, it was decided to redo the steps of model selection, training og evaluation for both the random forest (RFC) and neural network (NN) using chronologically split data. This time the January-August 2022 data was included in the entire process together with the original dataset for 2020-2021, resulting in even more samples for the training process. The portion of data used for training was still 60% as described in 6, and 20% for the validation and test dataset each. The redo was only performed on F01P01 and B01P01 in the training of a random forest classifier and a neural network (for reasons presented in the following section), thus the feature selection presented in sec. 11 was reached using the randomly split data, and the datasets referred to in this section is therefore randomly split. When discussing the results for RFCs and neural networks for F01P01 and B01P01 specifically, it is the chronologically split data that is referred. However, in any instances where these two datasets are compared to any of the other datasets (i.e F01P02, F02P01, F02P02, B01P02, B02P01, B02P02) it is the results from the randomly split data that is being discussed. It is recognized that this may not be optimal, however, it was not possible to redo all the steps using the chronically split data due to the time limitations, and the results obtained from the randomly split data can still provide important insight into the process of

using machine learning techniques in numerical weather prediction.

10 Structure of datasets and result selection

This thesis started by looking at many different datasets as described by table 2 in section 5.2. This included datasets of two different location, with two different sets of features for each of these two locations. In addition, these four datasets was then split into full forecast length, 0-6h, 6-12h and 12-66h forecast length. This was done in order to see how model performance would vary depending on how many features was included in the training process, as well as how the results differ with forecast length. In the course of this thesis, it became evident that some of these datasets was simply redundant, or that the process of splitting the data into different forecast lead times did not gain enough skill to compensate for the extra steps required to train the model for each forecast length. Because of this, the datasets used for each specific step within the machine learning process got smaller with each subsequent step, and in the end resulted in training an optimized neural network for F01P01 and B01P01 only (the datasets consisting of the full forecast lead time and MEPS variables only, overview of variables in tab. 3 in sec. 5.2). The process of narrowing in on the datasets used is illustrated in fig. 15, where the results of the datasets marked in gray is found in the appendix, while the ones in blue (F01P01, B01P01) is found in the main text.

11 Feature selection

11.1 Exploratory data analysis

Prior to doing the recursive feature elimination, some aspects of exploratory data analysis was done in order to highlight potential relationship within the data. Firstly, the Spearman rank between the variables and observed hourly precipitation was calculated using eq. 8. In addition, a base random forest classifier (*base* RFC) was trained on each dataset in its entirety (i.e no feature selection done), which then provided a feature importance calculated in the training process (as described in section 6.1). An exhaustive overview of the Spearman rank (r_{rank}) and feature importance found for the 01 and the 02 datasets can be seen in table B.3 and B.4 in App. B respectively. In this section, only the values for the selected features of F01P01 and B01P01 will be presented. In the two following subsections the rank matrix and hierarchical clustering will be presented in a brief manner. The results found from these exploratory data analysis will be further discussed in relation to the recursive feature elimination in the next section, 11.2.

Figure 16 shows the rank matrix for the 01 datasets of both Florida and Blindern. Equivalent figures for 02 can be seen in figure B.1 in App. B. In order to illustrate the order of the r_{rank} in the same figure as the rank matrix, the datasets have been sorted by descending Spearman rank before plotting the figure. Hence, when looking at the y-axis, the variables with the highest r_{rank} (in absolute terms) is at the top, while the lowest at the bottom.

Looking at figure 16a, one of the structures standing out is the blue area (equivalent to positive r_{rank}) in the upper left corner of the figure. This structure is made from the variables with the highest r_{rank} and include *precipitation, relative humidity, cloud fractions, zonal wind, CAPE, air pressure* and *surface downwelling shortwave radiation*. The fact that these variables are highly correlated with observed precipitation is not very surprising, considering the nature of them. *Zonal wind* is of importance because of the topography of Bergen and orographic enhancement. As expected, *air pressure* and *downwelling shortwave radiation* have a negative

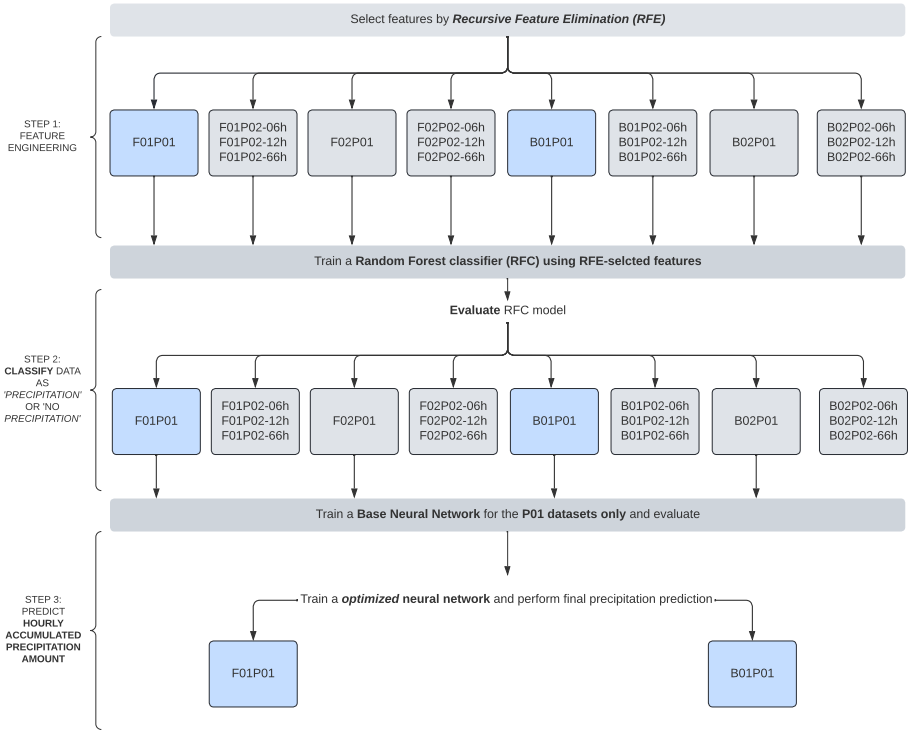


Figure 15: Diagram showing the structure of the datasets included in the thesis. The thesis starts with a broad set of datasets. Two distinct datasets with different variables, X01 and X02 (where X is either F - Florida or B - Blindern). In addition, these two datasets are then also split into different forecast period, referred to as full forecast-length (P01), 0-6 hours forecast lead time (P02-6h), 6-12 hours forecast lead time (P02-12h) and lastly 12-66 hours forecast lead time (P02-66h). All datasets are included in the first and second step, marked as *feature selection* and *classification*. From assessing the random forest classifiers (RFC), only the datasets with full forecast-length (i.e P01) is brought along to the third step of neural networks (NN). By looking at the performance of a *base NN* it was decided to only continue the training for F01P01 and B01P01 (both marked in blue), though with an inclusion of tendency variables. The results from these datasets marked in blue is presented in the main text below, while the results of the gray datasets is found in the appendix.

rank with the others, as well as with the observed precipitation. In addition to being highly correlated with observed hourly precipitation, these variables also have a rather high rank among themselves, indicating redundant information among the lot of them.

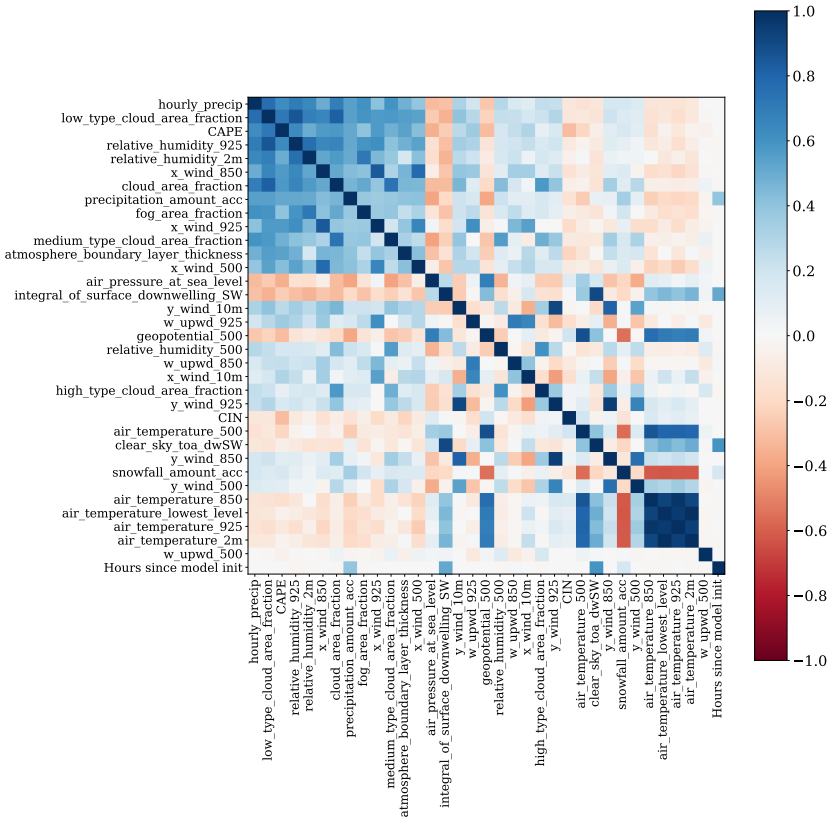
Figure 16b shows the rank matrix for Blindern, also sorted by descending Spearman rank. Most of the variables found relevant for precipitation for Florida, is the same for Blindern. The order of the Spearman rank of these variables vary somewhat, however, forecasted *hourly precipitation* and *low type cloud area fraction* proves to be ranked highest for both locations. One thing to note is that while *zonal wind* has a high r_{rank} for Florida, the *meridional wind* is more important for Blindern, although at an overall lower rank than what we see for Florida, indicating that it is of greater importance for Bergen, due to the prevalence of orographic enhancement from the forced lifting of the moist westerly air.

Figure 17 shows the hierarchical clustering found using Ward's minimum variance method as described in section 3.3. From fig. 17a it is evident that the data for Florida can be clustered into five groups. The purple cluster is perhaps the most notable since it contains the variables with highest r_{rank} discussed above. Figure 17b shows the data is clustered into four groups for Blindern. In this case, the yellow cluster contains the variables with the highest r_{rank}, and contains more variables then the purple cluster for Florida. The fact that the variables with highest r_{rank} also show high rank among each other may explain why they are grouped together in the hierarchical clusters. The implications of this will be further discussed in the following section.

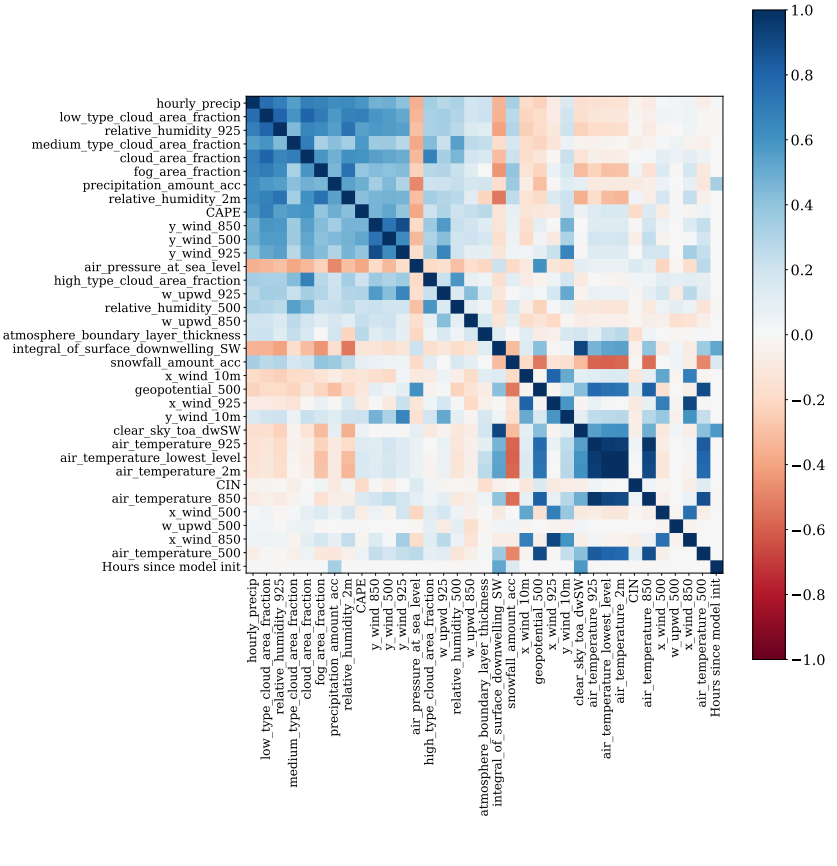
11.2 Recursive feature elimination

Because the optimal number of features for each dataset is unknown, sklearn's recursive feature elimination with a 5-fold cross-validation (RFECV) was used, as described in sec. 6.1. The RFECV was used on all datasets, and in some instances the optimal number of features (from this point onward refereed to as *n features*) that were selected was notably higher than for equivalent datasets. This was seen for F01P01, F01P02-66h and F02P02 (all forecast-lengths), and tab. 10 shows an overview of the *n features* for all datasets. To investigate this further, the objective used by the RFECV (here, balanced accuracy, see method 7.1 for further details) was plotted against *n features* in order to have a closer look at how the performance of the model behaved in relation to the number of features. These plots showed that for the instances where a higher *n features* was selected as the optimum, the objective had in fact stabilized at a lower number and the RFECV select the higher number because this provided the the global maximum. In these cases, sklearn's RFE was used as a second step, with *n features* set to a reasonable number found by manually investigating the plots in question, as well as looking at the number chosen for equivalent datasets. Figure 18 show such a plot for F01P01, and show how there is a steep increase in the objective going from one to four variables, before a slower increase as you reach ten variables. The RFECV selected *n features* to be 20, however, it is clear from the plot that there is little difference in the objective of 10 variables compared to 20. Also taking into consideration that for the equivalent dataset for Blindern the RFECV found *n features* to be 9, it was reasonable to use the RFE for F01P01 with *n features* set to 10.

Figures showing plots for F01P02-66h and F02P02 can be seen in fig. B.4 in App. B.2. The plot for F01P02-66h seen in fig. B.4a shows similar characteristics as the plot for F01P01, and RFECV found *n features* to be 20 also in this instance. The objective seems to stabilize at an even lower number than for F01P01, however, it is not a significant difference, and it was therefore decided to run the RFE with *n features* set to 10 also in this case. Figure B.4b and B.4c show the plots for F02P02-6h and F02P02-12h respectively. Both of these show similar plots, with a small increase in objective after 10 features. Comparing to the number found for

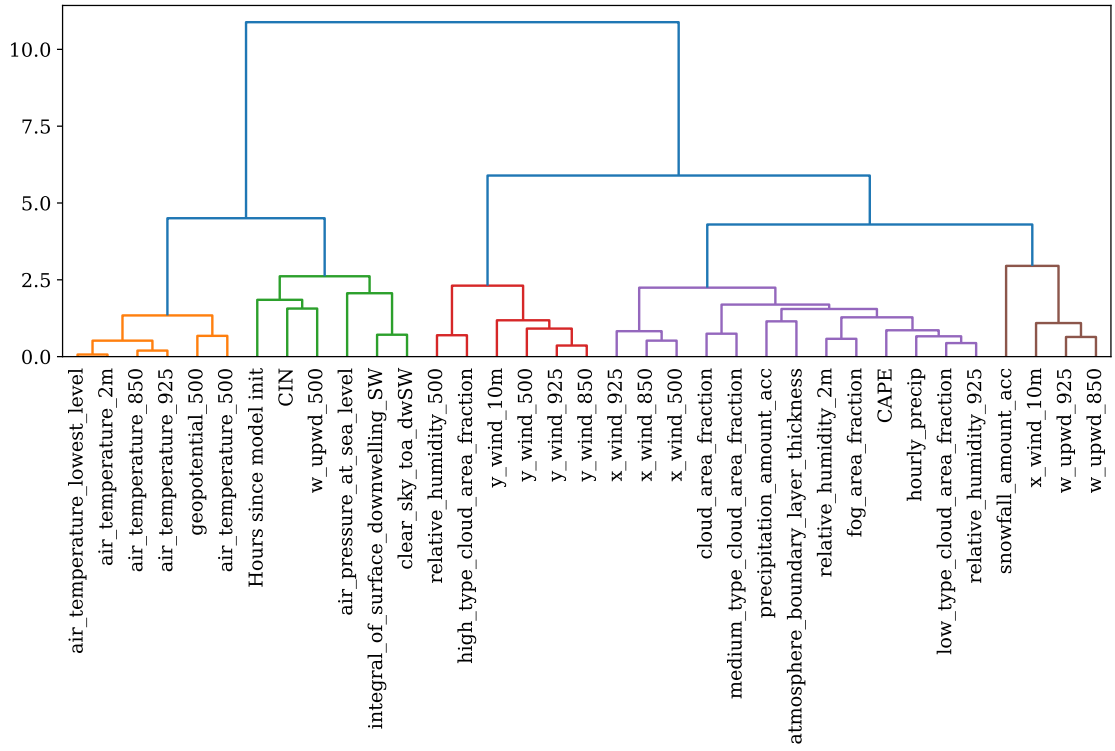


(a)

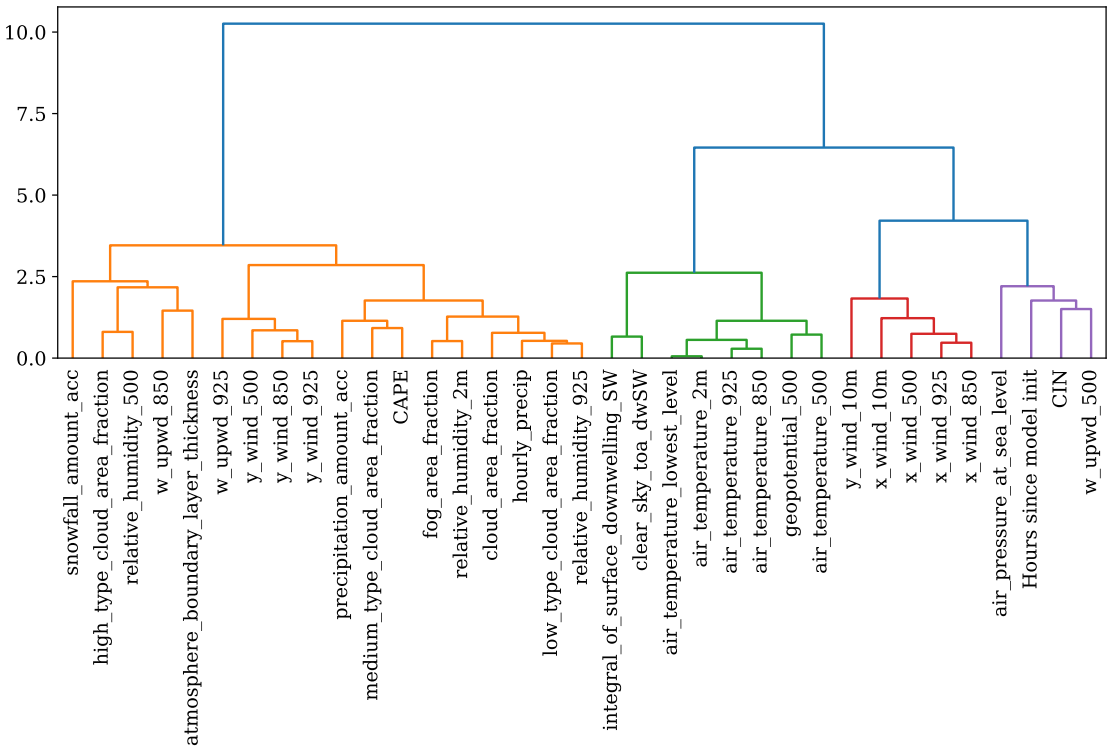


(b)

Figure 16: Figure shows the Spearman rank matrix for the 01 datasets for (a) Florida (F01) and (b) Blindern (B01). The datasets have been sorted by descending Spearman rank (r_{rank}) before plotting.



(a)



(b)

Figure 17: Figure show a schematic representation of the hierarchical clustering for the 01 datasets. Fig. 17a shows the clustering for Florida (F01), where the purple contains the variables with the highest r_{rank}. Fig. 17b shows the equivalent for Blindern (B01), where the yellow cluster contains the variables with the highest r_{rank}.

similar datasets such as F02P01, B02P01 and B02P02-66h, it was decided to run the RFE with *n features* set to 30. The same was done for F02P02-66h (fig. B.4d)

One interesting finding is the general low number selected for B01P02-66h and B01P02-12h. Considering the low percentage of the samples that actually can be classified as *precipitation*, it is rather surprising that the RFECV only select one and two features (*hourly precipitation* for both, and in addition *accumulated precipitation amount* for B01P02-12h). An initial thought may be that the random forest only need MEPS precipitation forecast to do adequate and reasonable classifications of the data. However, these numbers are vastly different for any similar datasets, which raises the question whether the combination of significantly fewer samples and the low percentage of precipitation occurrences within the observation, actually sway the model towards classifying most samples as class 0, *no precipitation*. Although not done in this thesis, this could be investigated further by looking at the model evaluation for each class individually.

Table 10: Optimal number of features, n, found by RFECV for each dataset. The n found for F01P01 and F01P02-66h is considerably higher than the n found for similar datasets. The n found for F02P02 is many times higher than those found for B02P02.

	n features		**n features**		**n features**		**n features**
F01P01	20	**F02P01**	30	**B01P01**	9	**B02P01**	29
F01P02-6h	8	**F02P02-6h**	50	**B01P02-6h**	1	**B02P02-6h**	1
F01P02-12h	11	**F02P02-12h**	71	**B01P02-12h**	2	**B02P02-12h**	9
F01P02-66h	20	**F02P02-66h**	67	**B01P02-66h**	9	**B02P02-66h**	25
FB	24						

Table 11 shows and overview of the selected features for each of the datasets F01P01 (left) and B01P01 (right), and their respective r_{rank} and feature importance. The ten features selected by RFE for F01P01 includes *precipitation, cloud and fog area fraction, relative humidity, CAPE* and *zonal wind* at 925 hPa and 850 hPa. This is very much in agreement with the variables with the highest r_{rank} as discussed in sec. 11.1. Comparing r_{rank} and feature importance for the selected features it is noteworthy that the three features with the highest r_{rank}, specifically *hourly precipitation*, *low type cloud fraction* and *CAPE*, also is assigned the highest feature importance by the *base* RFC.

The features selected for B01P01 are many of the same as for F01P01. As was evident from the order of the r_{rank}, *meridional wind*, although only at 850 hPa, was selected instead of zonal wind. The selected features with the highest r_{rank} also corresponds to those assigned the highest feature importance. The overall good agreement between the Spearman rank and the assigned feature importance from the *base* RFC (random forest classifier) indicates that the features selected by the RFECV/RFE algorithm are indeed reasonable. It is however interesting to note that the RFECV/RFE selected features were grouped together in the Dendrogram and also showed rather high rank among themselves (fig. 17 16). This of course raises the question whether some of these features are redundant. This was only briefly looked at for F01P01, and will be presented only as an anecdote in the following description. Several types of features was evident from the RFECV selected data, and could be categorized into precipitation, cloud area fractions, wind and humidity. One category of features (for example the different cloud area fractions) was removed from the selected set of features, and then each individual variable of this type was reintroduced to the dataset separately to assess the model objective. The variable that gave the best objective was then kept in the dataset, while the others were discarded. This was repeated for all the different categories, resulting in a quite small dataset. This yielded poorer results than including all of the selected features found from RFECV, and this process

was thus not repeated for any other datasets. Therefore, although there might be redundant information across the selected features, it seems like they each contribute some information gain which collectively adds up to a better model performance, though this could use further research, as well as how much change in model performance one will tolerate in the exchange for a more simplified model.

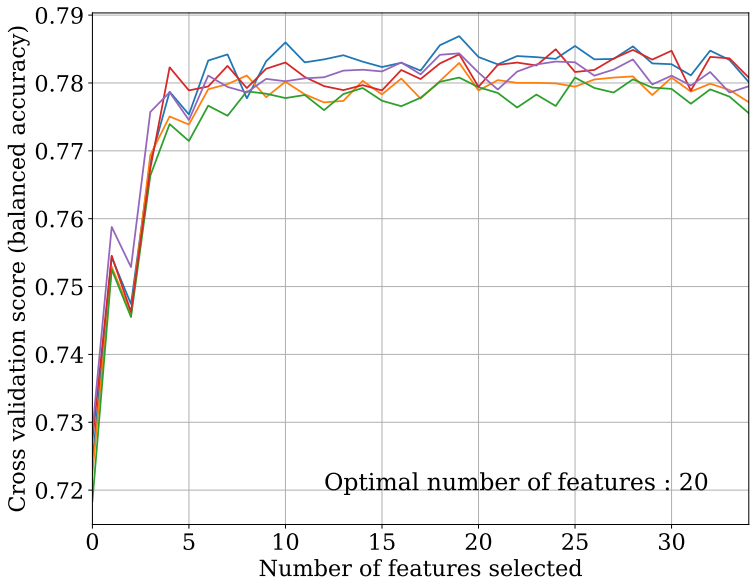


Figure 18: Figure shows balanced accuracy plotted against number of features, n for F01P01. Each differently coloured plot represents one of the 5-fold cross-validation used by the RFECV.

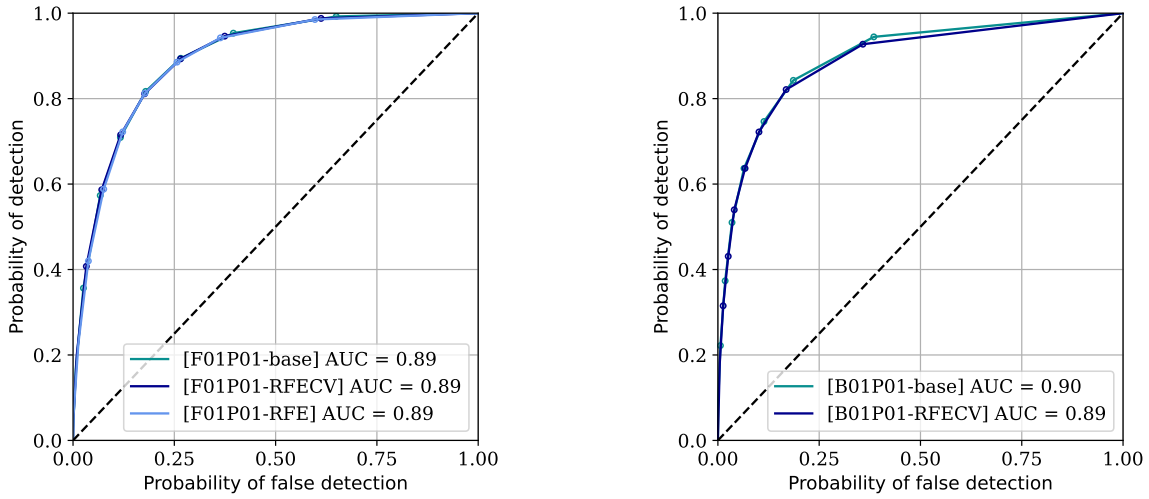
Table 11: An overview of the *feature scores* of the RFE-selected features for Florida (F01P01) on the left and the RFECV-selected features for Blindern (B01P01) on the right. The term *feature scores* is here referencing the Spearman rank, r_{rank}, and *feature importance* obtained from the *base* random forest, which is is informative of the total reduction in the Gini Index brought by the feature [Scikit-learn User-Guide, 2022, 1.11.2.5]. (Accumulated) *hourly precipitation* and *low type cloud area fraction* has the highest r_{rank} and *feature importance* for both Florida and Blindern.

F01P01			**B01P01**		
RFE	r_{rank}	**Feat. imp.**	**RFECV**	r_{rank}	**Feat. imp.**
precipitation_amount_acc	0.449	0.043	**hourly_precip**	0.52	0.199
low_type_cloud_area_fraction	0.53	0.174	**relative_humidity_925**	0.419	0.139
x_wind_850	0.468	0.098	**medium_type_cloud_area_fraction**	0.409	0.09
fog_area_fraction	0.438	0.042	**cloud_area_fraction**	0.408	0.071
cloud_area_fraction	0.463	0.01	**fog_area_fraction**	0.389	0.043
medium_type_cloud_area_fraction	0.424	0.027	**precipitation_amount_acc**	0.385	0.037
x_wind_925	0.424	0.074	**relative_humidity_2m**	0.362	0.062
CAPE	0.495	0.101	**y_wind_850**	0.352	0.057
relative_humidity_2m	0.49	0.086	**low_type_cloud_area_fraction**	0.438	0.14
hourly_precip	0.561	0.121	**-**	-	-

11.2.1 Evaluation of the feature selection

To summarize the performance of the different models run during feature selection (base RFC, RFECV, and in some cases RFE) one can look at the calculated AUC and visualize how performance depend on threshold probability by looking at the ROC curve (method sec. 2.6). Figure 19 shows the results from models trained on different feature selections of F01P01 and B01P01. Looking at the ROC curves for Florida in fig. 19a it is clear that there is no difference between *base* RFC, RFECV or RFE. The fact that the model shows no difference in performance with 30 or 20 features compared to 10 features is a testament to the fact that there is much redundant information within the dataset that is not utilized by the machine learning algorithm. Figure 19b shows insignificant difference between 33 and 9 features, which further supports this claim.

Figure B.5 in B.5 shows ROC curves for all datasets, with Florida on the left and Blindern on the right. To summarize the main findings from these ROC curves is that there is little difference between performance of the models trained on F01/B01 and F02/B02, suggesting that the feature selected by the RFECV/RFE contains enough information for the RFC to classify the data. A general pattern for the P02 datasets is that the model performance is greatest for 0-6 hours, then declining with each forecast-length. However, the difference between model performance for P01 versus P02 is not very significant.



(a) **(b)**

Figure 19: Figure shows ROC curve for all models trained for F01P01 and B01P01 during feature selection. Note that these models are evaluated on the *validation* data used specifically for model selection (and *not* the *test* data intended for the final model evaluation) (a) ROC curves for *base* RFC, RFECV and RFE trained on F01P01. (b) ROC curves for *base* RFC and RFECV trained on B01P01.

Table 12 shows an overview of the verification metrics for the different model in the feature selection process for F01P01 and B01P01. It is important to note that this evaluation was not done on the *test* data, but the *validation* data, and as the models trained in the feature selection process is not the final random forest classifier (RFC), the verification metrics for these models will not be discussed in detail. However, the take-away from this evaluation is the fact that the performance is seemingly unchanged when reducing the datasets to approximately 10 features, emphasizing the necessity to do a proper feature selection. Though keeping redundant features does not appear to make the RFC performance any poorer, this is likely due to the fact that random forest is quite robust against overfitting, and this will likely not be reflected in other machine learning algorithms. Table B.8 and B.9 in App. B.6 shows an exhaustive overview of the verification metrics for all Florida and Blindern datasets respectively.

Table 12: Table shows the verification metrics for all models run during feature selection for F01P01 in the upper panel and B01P01 in the lower panel, with the abbriviations defined as ACC: accuracy, POD: probability of detection, POFD: probability of false detection, FAR: false alarm ratio/rate, SR: success rate, CSI: critical success index, SS: skillscore. The *base* is the *base* classifier trained on all features in the datasets, while *RFECV* is the *base* RFC trained on the RFECV-selected features (selected by the recursive feature elimination cross-validation module, i.e RFECV), and the RFE is the *base* RFC trained on the RFE-selected features (selected by the recursive feature elimination module, i.e RFE).

		Base	RFECV	RFE	MEPS
F01P01	**ACC**	0.84	0.84	0.83	0.79
	POD	0.65	0.66	0.66	0.5
	POFD	0.09	0.09	0.1	0.09
	FAR	0.26	0.27	0.27	0.32
	SR	0.74	0.73	0.73	0.68
	CSI	0.53	0.53	0.53	0.4
	SS	0.42	0.42	0.41	0.26
	Bias	0.88	0.9	0.91	0.73
B01P01	**ACC**	0.9	0.9	-	0.89
	POD	0.44	0.49	-	0.56
	POFD	0.03	0.03	-	0.05
	FAR	0.26	0.29	-	0.37
	SR	0.74	0.71	-	0.63
	CSI	0.38	0.41	-	0.42
	SS	0.29	0.29	-	0.23
	Bias	0.6	0.68	-	0.9

Both Spearman rank and feature importance line up well with the features selected by the recursive feature elimination. It is however interesting to see that so many similar features are selected. For example, *low type cloud area fraction*, *medium type cloud area fraction* and *cloud area fraction* are all selected for both locations, which seems redundant. The same is true for *zonal wind* at both 925 hPa and 850 hPa for Florida. Because of the seemingly redundancy of some features, further work could include more feature selection and the use of domain knowledge to further reduce redundant information within the dataset. Additional feature engineering could possibly provide more compressed information than the ones included in this work and thus simplify the dataset even more. The zonal wind at 925 and 850 hPa selected for Florida could for example be replaced by one variable either by aggregating the two levels or by calculating the wind normal to topography. The high rank among the features shown in fig. 16 and the grouping seen in fig. 17 show strong indication that the feature selection could be optimized even further. Nevertheless, due to time restriction it was decided to stick with the RFECV/RFE selected features when continuing onto the next part of model selection in the RFC process.

From this point forward it should be understood that when referring to the different datasets, for example F01P01, it is the RFECV/RFE selected features only that is referred. In any case where it is the full original (pre-selected) dataset that is referenced, this will be explicitly stated.

12 Classification of *no precipitation/precipitation* using random forest

This part of the thesis covers the machine learning process of the classification task. Each sample will be assigned one of two classes, which are categorized by *hourly accumulated precipitation amount*, $RR < 0.1mmh^{-1}$ (*class 0; no precipitation*) and $RR \geq 0.1mmh^{-1}$ (*class 1; precipitation*).

This is achieved using a random forest classifier as described in section 7 and the results will be presented in the following structure; Firstly, model selection to optimize the model structure for the given task, secondly, fitting the optimized model to the training data and lastly, evaluate model performance using previously unseen *test* data.

12.1 Model selection

Optimizing model performance is done by tuning different hyperparameters. The tuning is done using the GridSearchCV as described in sec. 7.1. Table 13 shows the hyperparameters chosen by the grid-search for F01P01 and B01P01. The grid-search selected shallower trees and fewer trees in the forest (estimators) for Florida than for Blindern. This is possibly due to the fact that there is significantly less class 1 (*precipitation*) samples for Blindern (i.e lower percentage of precipitation within the Blindern data), meaning that deeper trees and more estimators are required to be able to solve the task sufficiently. On the other hand, this is not reflected in the grid-search done for the randomly split data and is thus only speculative. In addition, it is possible to achieve similar results with many different combinations of hyperparameter-settings, making it difficult to assess individual hyperparameters.

An extensive overview of the hyperparameters selected by the grid-search for the P01 and P02 data (randomly split) can be found in C.11 and C.11 in App. C.1. From this overview it is evident that no pruning (*ccp_alpha* = 0.0) is selected for all P01 cases (full forecast lead time), while there seem to be equal split between 0.0 and 0.150 for P02 (split into forecast lead time). A general pattern seems to be that pruning is selected for P02-6h and P02-12h, while less for P02-66h. This is possibly due to the fact that the P02-6h and P02-12h contain far fewer samples than P02-66h (which is closer to the amount of samples in P01). Fewer samples makes it harder for the random forest to generalize, making the model more prone to overfitting, thus pruning might be chosen to minimize this. Another way to minimize overfitting is to reduce the depth of the trees. Shallow trees are favored for all datasets, however, despite there being fewer samples for the P02-6h and P02-12h datasets, the grid-search seem to favor even shallower trees for these than for the larger datasets, which also might be a way to reduce the chance of overfitting. For P01 there seems to be a preference for the class weight (described in tab. 7 in sec. 7.1) to be set to *balanced_subsample*, while it is an equal split between *balanced* and *balanced_subsample* for P02. The option *None* was not selected in any of the cases, which makes sense considering how unbalanced the classes are. The grid-search prefers the depth of the trees to be more on the shallow end for both types of datasets. The number of estimators (described in tab. 7 in sec. 7.1) do vary, though 250 estimators are selected most frequently.

Table 13: Hyperparameters selected in the grid-search done by the GridSearchCV for F01P01 and B01P01.

	ccp alpha	**class weight**	**criterion**	**max depth**	**n estimators**
F01P01	0.0	balanced	gini	10	50
B01P01	0.0	balanced	entropy	25	250

After the grid-search had been completed, the optimized model's (from this point referred to as the GridSearchCV) performances was evaluated and compared to MEPS and the *base* RFC (random forest classifier), which in this section will also be referred to as reference models. The results for the GridSearchCV for F01P01 can be seen in table 14. Though probability of detection (POD) is significantly better than both the *base* RFC and MEPS, this is followed

by an significant worsening of probability of false detection (POFD) compared to the reference models. The GridSearchCV also has slightly lower skill (SS) then the *base* RFC, though it scores higher than MEPS. This, together with an increase in bias, creates a picture of over-predicting precipitation-events. The GridSearchCV does detect more of the actual precipitation events, but it also overestimates the occurrence. According to this, the *base* RFC provides overall better performance, and is thus a better model choice than the one selected by the grid-search.

Looking at the model selection results for B01P01 a similar pattern of an increased POD with an accompanying significant increase in POFD is found. A decrease in success ratio (SR) and critical success index (CSI) as well as a significant increase in bias results in an overall worse skill than both reference models. In fact, the SS is negative, meaning it performs worse than just guessing the majority class for each sample. In contrast to F01P01 however, the *base* RFC for B01P01 has a lower POD than MEPS, yet a comparable POFD. This is accompanied with a slight increase in SR and CSI, and a more significant lowering in bias, resulting in a skill comparable to MEPS. Nevertheless, it seems the *base* RFC is not entirely able to identity the precipitation events with the minority class making up such a small percentage of the dataset.

Table C.13 and C.14 in App. C.1.2 gives a exhaustive overview of the model selection results of P01 and P02 respectively (random split). The most unexpected result (from both splits) is that the models selected by the grid-search consistently perform worse than the *base* RFC used in the feature selection. What is perhaps even more noteworthy is the fact that the hyperparameters-settings used in the *base* RFC is included in the grid-search, meaning the grid-search evaluates this combination and deems it as less optimal than the one it selects. In other words, the *base* RFC score lower on the evaluation method used by GridSearchCV, balanced accuracy, than the model selected by the grid-search. It is nothing new that accuracy is not a sufficient measure of model performance when class distribution is unbalanced, as the model can score very well by just predicting the majority class every time. However, the *balanced accuracy score* takes the imbalance into consideration by applying class-balanced sample weights [Buitinck et al., 2013]. Because F_1 (described in method sec. 7.1) is another popular choice for imbalanced data, a grid-search using F_1 as objective was done on F01P01 in order to assess how this compares to the outcome of using the balanced accuracy score. The skill score of the grid-search using F_1 was calculated to be 0.33, which is exactly the same skill score as for the grid-search using balanced accuracy (both calculated using the randomly split data). Because of this, balanced accuracy was kept as the objective for the grid-search.

These results indicate that the methods used for evaluating classification models does not translate that well into the methods used to evaluating numerical weather prediction. In this case, the selected model got the highest score at the classification score, but got lower scores on the forecast verification metrics compared to the *base* RFC.

Furthermore, in order to investigate the possibility of the feature selection that was done influencing GridSearchCV performance negatively, a grid-search was done on the original P01 (containing all features) datasets as well. This gave equivalent results, both concerning hyperparameter tuning and verification metrics for the selected models. The exact results will not be discussed further, but can be found in tab. C.13 in App. C.1.2. This indicated that feature selection has no significant impact on the hyperparameter tuning, and thus this process was not repeated for the P02 data.

Table 14: Verification metrics for the model selection of the random forest classifiers (RFC) for F01P01 (upper) and B01P01 (lower), in comparison to MEPS. The models are trained on the selected features found in sec. 11, and evaluated on the *validation* data. The *GridSearchCV* references the RFC model with the hyperprameter-settings found in the grid-search by the GridSearchCV algorithm (as presented in tab. 13). The *Base* is the base RFC used in the feature selection (settings specified in tab. 6 in sec. 6.1.2). The *Base* RFC consistently outperforms the *GridSearchCV* RFC for both locations.

		ACC	POD	POFD	FAR	SR	CSI	SS	Bias
F01P01	**GridSearchCV**	0.76	0.92	0.35	0.35	0.65	0.61	0.42	1.41
	Base	0.79	0.75	0.18	0.26	0.74	0.6	0.49	1.01
	MEPS	0.74	0.57	0.14	0.26	0.74	0.47	0.37	0.77
B01P01	**GridSearchCV**	0.84	0.88	0.16	0.59	0.41	0.39	-0.37	2.13
	Base	0.92	0.5	0.03	0.31	0.69	0.41	0.28	0.72
	MEPS	0.91	0.61	0.04	0.36	0.64	0.45	0.26	0.95

From the results presented above, it is clear that *sklearn's GridSearchCV* does not provide the optimal model for the intended use in this thesis. This highlights the implication that the typical objectives used in classification tasks does not reflect the methods used in forecast verification. In this study, the *base* RFC utilized in the feature selection provided overall better results. Therefore the GridSearchCV models were at this stage discarded, while the *base* RFC was selected as the final classifier. Thus, the *base* RFC will from this point forward be referred to as just the RFC.

12.2 Model evaluation

The final step in this process was then to evaluate the random forest classifier's (RFC) performance. In order to do this, the selected RFC were set to predict the class of *unseen* data, i.e the *test data* described in section 4, spanning mid February to the beginning of August 2022.

From table 15 we see that the random forest classifier (RFC) for F01P01 are able to increase the POD (probability of detection) with 16%, but also increases the POFD (probability of false detection) with 10% compared to MEPS. The other verification metrics are comparable to MEPS, resulting in a 16% *increase* of skill for the RFC. The ROC curve in fig. 20a very much resembles the ROC curve plotted during feature selection (randomly split), which also got $AUC = 0.89$. From the reliability diagram shown in fig. 20b we see that the RFC overestimates the probability of precipitation for observations with an observed frequency ≤ 0.7, meaning the RFC has a tendency of overpredicting.

The model evaluation for random forest classifier (RFC) for B01P01 (also tab. 15) indicate that the RFC is not able to identify precipitation events to the same degree as MEPS (RFC has 24% lower POD than MEPS) and this is supported by the ROC curve in fig. 21a. Though the RFC accuracy is comparable to MEPS, this is likely due to the high occurrence of no precipitation-events. The CSI (critical success index) of the RFC is 15% lower than MEPS. The POFD is also significantly lower for the RFC than MEPS (50%), which is a good thing. However, together with the low POD this is likely be due to a systematic under-prediction of precipitation events due to a too small percentage of precipitation-events within the training data. Because of this it is strange to find that the skill score for the RFC is ten times higher than MEPS, even with the 50% reduction in POFD. In the model selection, where the models was evaluated on the validation data (see tab. 14), the SS of the RFC is only 7% higher than MEPS, which seems more realistic.
The low POD and POFD might indicate a systematic underprediction, but this is on the other

hand not supported by the reliability diagram in fig. 21b which rather indicate overprediction due to the graph falling under the one-to-one line. This is somewhat contradictory to the verification metrics though, which indicate underprediction with bias$\leq$ 1, and the cause of this discrepancy is not clear. Note that MEPS is *unbiased* (bias equal to 1).

The verification metrics for all the random forest classifiers (RFC) are found in table C.15 in App. C.1.3. With exception to the RFC for the Blindern P02-6h datasets, all other RFC perform significantly better than MEPS. Even though the skill of MEPS is quite comparable for Blindern and Florida, there is a notable difference in the skill of the RFC for the different locations. The Blindern RFC has overall a smaller gain in all metrics compared to MEPS, than the Florida RFC. However, considering the minority class (i.e *class 1: precipitation*) only makes up 14% of the data for Blindern, compared to 32% for Florida, this difference is expected. This demonstrates how important the quantity of data is in the training process, and how difficult it is to train on very unbalanced data. Although it was decided not to do in this study, one could try correcting the unbalance with either up- or downsampling to see whether this could bring the performance of Blindern to the level of Florida. One reason undersampling was decided against was to keep as many samples as possible, as we know the machine learning algorithms perform better the more data is available. Upsampling was decided against due to the fact that one only generate more samples from already existing data, and thus not actually adding any new information to the dataset. Nevertheless, in further work it could prove useful to research how up- and downsampling could contribute to the overall model performance, particularly for Blindern.

An interesting result is the fact that there is no notable gain when training a separate model for the different forecast intervals (i.e 0-6h, 6-12h, 12-66h). Because the forecast skill goes down as the forecast time increases, one reasonable hypothesis is that there would be a greater gain of skill for the 6h and 12h datasets, compared to the skill gain of the full forecast period. On the other hand, the amount of data the random forest classifiers (RFC) has to train on is greatly reduces for these smaller datasets, which one can expect leads to a reduced gain of skill. Table C.15 in App. C.1.3 show that both the RFC and MEPS achieve the greatest skill for P02-12h, followed by P02-6h, then lastly P02-66h, just as expected because of chaos. However, when comparing the respective P02 RFC with the corresponding MEPS, it is evident that the greatest *increase* of skill is P02-66h (50% increase for F01P02-66h and 33% increase for B01P02-66h compared to MEPS), followed by P02-12h, and lastly P02-6h. There is a notable difference between the locations in this. Although F01P02 (-6h, -12h, -66h) achieves an increase of skill of 33%, 44.8%, 50% compared to MEPS, which is a significant improvement, when averaged it is equivalent to F01P01s increase of 42%. B01P02-6h on the other hand achieve a 4% *decrease* in skill compared to MEPS, while B01P02-12h and B01P02-66h achieve 11% and 23% increase respectively. This averaged is however not comparable to B01P01 which performs significantly better with a 25% increase of skill in comparison.

From these results it is evident that the loss of data in the training is especially severe for Blindern, resulting in an overall poorer performance of B01P02 compared to B01P01. Even though F01P02-12h and F01P02-66h score better than F01P01, the loss of data results in significantly poorer performance for F01P02-6h (compared to F01P01), and thus the loss of data samples in the training counteract any gain of shorter forecast lead times. In addition, training a separate model for different forecast periods complicates the process of implementing machine learning in the steps of numerical weather prediction. Not only could it prove necessary to do feature selection multiple times, one does also need to do model selection and train several models. Furthermore, having several distinct forecast periods can also lead to discontinuity going from one period to the next, meaning even more post-processing is needed. Taking all of this

into account, this study supports that it is sufficient to train *one* model for the entire forecast period (0-66h in this case using AROME-MEPS).

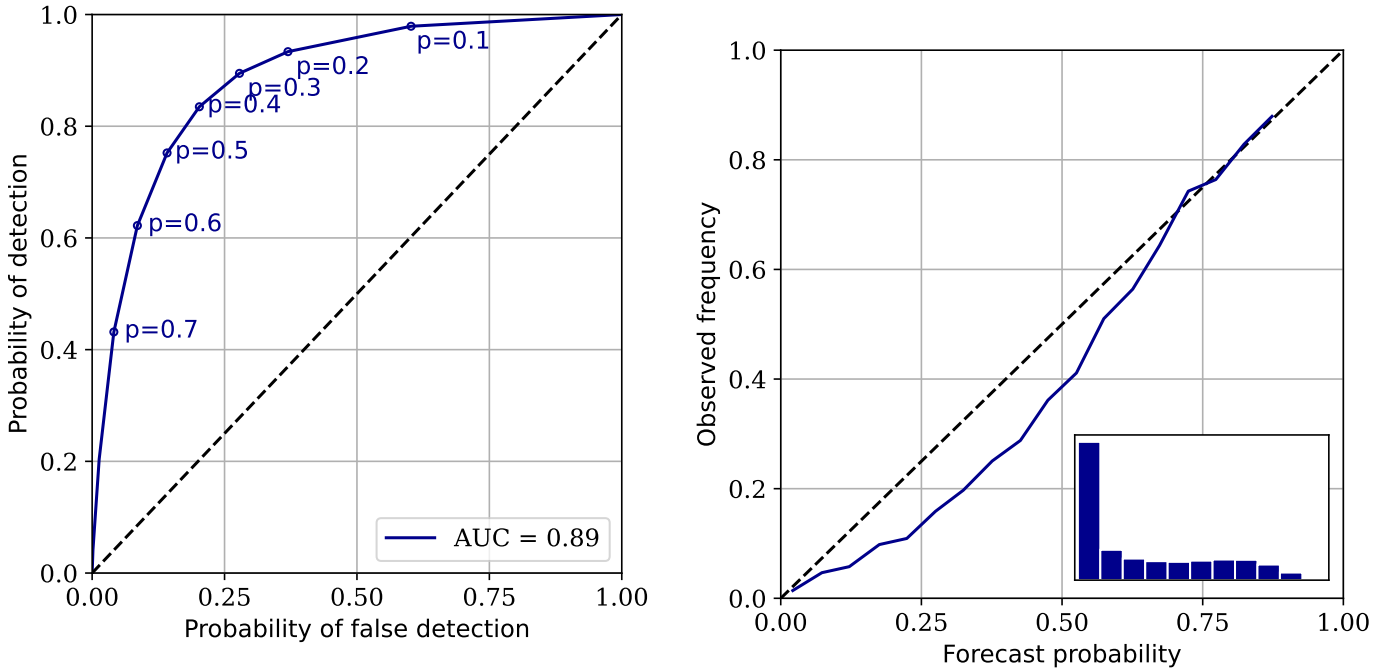
When comparing the skill score found with the chronologically split data with the one found with the randomly split data (which can be seen in tab. C.15), it demonstrates why it proved necessary to redo the process with chronologically split data. It should be noted that the data from the two splitting methods span different time periods (January 2020 to January 2021 for randomly split, January 2020 to July 2022 for chron. split), the MEPS results between the two spits are comparable, supporting the claim that the difference in skill score between the splitting methods is in fact due to overly positive results for the randomly split, stemming from cross-contamination between the training and test set from auto-correlation within the data.

One last point is that the final performance of the random forest classifiers (RFC) on the unseen (randomly drawn) test data very much reflect the values presented for the validation data. In contrast, comparing the results in tab. 14 and 15 from the chronologically split we see quite a drastic difference, for both the RFCs and MEPS. The latter contains an uneven distribution of precipitation data across the training, validation and test data due to the limited time-period, and this unevenness becomes evident when comparing the validation and test performance. The test data is from mid February 2022 until August 2022 and thus span the summer months that often prove more challenging to forecast in terms of precipitation, and this is likely an important factor in why there is such a big difference in performance across the validation and test data.

Table 15: Verification metrics for the final random forest classifiers, which in the model selection was determined to be the *Base* RFC from tab. 14. The performance of the RFC is evaluated on the *test* data spanning the period 18th February 2022 16:00 UTC until 3rd August 12:00. The results for F01P01 (upper) and B01P01 (lower) is compared to MEPS.

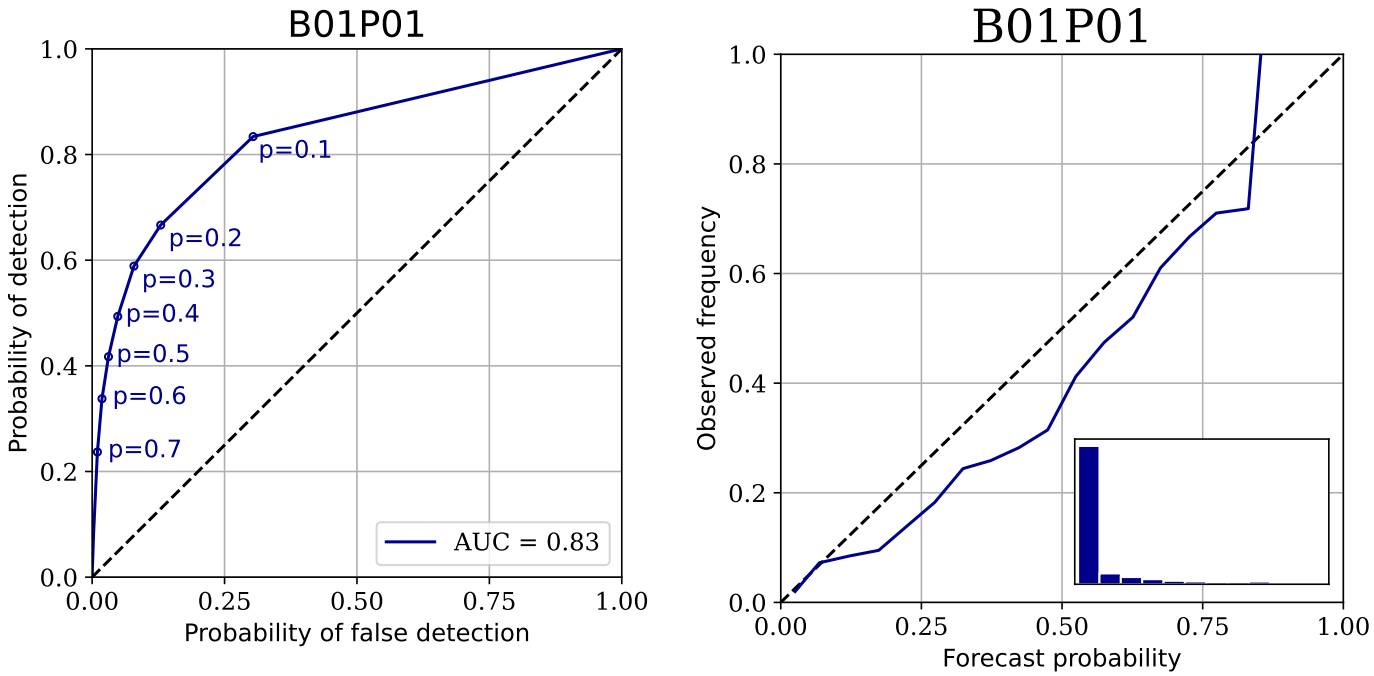
	ACC	**POD**	**POFD**	**FAR**	**SR**	**CSI**	**SS**	**Bias**
F01P01	0.85	0.69	0.11	0.38	0.62	0.49	0.28	1.11
MEPS	0.84	0.59	0.1	0.37	0.63	0.44	0.24	0.95
B01P01	0.93	0.38	0.02	0.42	0.58	0.29	0.1	0.65
MEPS	0.92	0.5	0.04	0.5	0.5	0.34	0.01	1.0

The figures in App. C.1.3 shows an exhaustive overview of the ROC curves and reliability diagrams for all datasets (randomly split). The reliability diagrams of the P01 random forest classifiers (RFC) show the models have good calibration, with a tendency to underpredict the cases of higher observed frequency. The reliability diagrams for the P02-RFCs shows more variability, especially on Blindern, due to the smaller amount of samples available for each forecast period. Comparing the diagrams from the chronologically split data with the randomly split shown in App. C.1.3, we see a significant difference. The reliability diagrams showed good calibration for the randomly split data, while we see a pattern of overprediction for both F01P01 and B01P01 of the chronological split. Although the ROC curve for F01P01 (fig. 20a) is comparable to the one found with randomly split, the ROC curve for B01P01 (fig. 21a) shows a significant reduction in performance compared to the randomly split.



(a) (b)

Figure 20: Left figures shows the ROC curve for the final RFC for Florida (F01P01) evaluated on the *test* data spanning the period 18th February 2022 16:00 UTC until 3rd August 12:00. Right figure show the reliability diagram, consisting of the reliability graph and rank histogram (see 8.3 for further explanation)



(a) (b)

Figure 21: The same as figure 20, but for Blindern (B01P01).

While the grid-search selected the hyperparameters that got the best evaluation score, when calculating the different evaluation metrics used in numerical weather prediction it was evident

that these models were far from optimal. The results of the grid-search shown in table 14 shows the GridSearchCV performed significantly worse compared to the *base* RFC used in the feature selection. It is clear that the objective traditionally used for classification tasks does not translate well to the verification methods used in numerical weather prediction. The RFC used in this thesis is thus not optimized, and it is likely that better results would be achieved if a manual grid-search using our verification methods. These results indicate that model performance is to a surprisingly high degree reliant on the hyperparameters used. Based on the hyperparameters used in the *base* RFC, one can speculate that a strict and shallow tree performs better as it might be less prone to overfitting, and thus does a better job at generalization.
From the results presented in table C.15, training separate models for different forecast lead times (P02-6h, P02-12h, P02-66h) seems to be an unnecessary step. Considering the added steps in the machine learning process when utilizing multiple models, as well as the effect of fewer data samples in the training, it can be concluded that training *one* single model for the entire forecast period is to be preferred. Therefore, it is only the datasets with the full length forecast period that will be utilized when training neural networks in the coming part.

Lastly, in the case of classification of *precipitation* and *no precipitation*, there is for the most part little difference in the RFCs trained on 10 features compared to the RFCs trained on 30 features, implying munch redundancies within the data. However, because of the different nature of regression task and neural networks, the 02P01 datasets was still advanced to the initial steps in the neural network process.

13 Predicting *hourly accumulated precipitation amount* using neural networks

In this section the results of using neural networks (as described in sec. 8) to solve the regression task of predicting *hourly accumulated precipitation amount* is presented and discussed.

13.1 *Base* NN and feature selection

The feature selection presented in sec. 11.2 was also used when solving the regression task using neural networks. A base neural network (from this point referred to as *base* NN) with the architecture described in section 8.1 was first trained using the previously RFECV/RFE-selected features only. In an effort to capture some temporal dependencies in the data, the *base* NN was then also trained on datasets where the tendency features highlighted in tab. 4 was included in addition to the selected features. Table D.18 in App. D shows the mean absolute error (MAE) of these models (note that these models are trained on the randomly split data). For class 0 (*no precipitation*) of F01P01 the MAE was reduced by 5% when including the tendency features, and class 1 (*precipitation*) had an 4.8% reduction. Interestingly enough, the MAE did not decrease for either classes of B01P01 when including the tendency features, and it was at first decided to not include the additional features when moving forward to selecting an optimized neural network. However, due to the fact that all other datasets had better performance with the tendency included, it was decided to train an optimized neural network for both the recursive feature elimination (RFE) selected variables only, as well as with the additional tendency features to assess whether an optimized neural network showed the same result as the *base* NN (concerning performance with and without the tendency variables). Although class 0 had the same performance with and without the tendency included, class 1 had an 8% decrease in the MAE with the tendency included compared to without. Based on these results, it was decided to include the tendency features in the further training of the neural networks for both F01P01 and B01P01.

13.2 Model selection

The model selection was done using KerasTuner as described in sec. 8.2. The architecture selected for each model is shown in tab. 16. The most striking difference between F01P01 and B01P01 is the number of hidden layers required to solve the task. While four hidden layers are selected for Florida (originally *one* hidden layer, described in next paragraph), Blindern require nine hidden layers. This is once again a testament to the difference in precipitation distribution within the data, where Blindern require a significantly deeper network in order to estimate the precipitation amount.

Originally, the grid-search for class 1 (*precipitation*) of F01P01 selected an architecture with only one hidden layer (see full architecture in tab. D.19 in App. D.3). However, after plotting the time-series of the neural network forecast of the test data (fig. D.10) it was very clear that there was an artificial cut-off of the hourly precipitation amount predicted by the neural network. This artificial cut-off was not evident in the predictions from the optimized neural network for the randomly split data. To look into this further, a neural network with the architecture found by the grid-search for the randomly split data (with 4 hidden layers) was trained on the chronologically split data. Table D.20 show the verification metrics for both neural networks and it is quite clear that there is a notable increase in performance, especially for the higher-intensity rain rates. Bias is overall better for with the 4-layer neural network, whilst skill score is better for the 1-layer neural network for rain rates $\leq 2.5mmh^{-1}$. For rain rates $\geq 2.5mmh^{-1}$ the 4-layer neural network has a higher skill score. Due to the generally better performance of the 4-layer neural network, it was decided to discard the 1-layer neural network at this point, and continue with the 4-layer neural network for class 1 of F01P01, and thus it is the latter that is presented in tab. 16.

Table 16: Table shows the architecture for each neural network of class 0 (*no precipitation*) and class 1 (*precipitation*), for each location. There is a clear difference of how many layers is needed for the two location to solve the task.

	Architecture of the network			
	Class 0		**Class 1**	
	Layer	**Units**	**Layer**	**Units**
F01P01	Input	96	Input	128
	Hidden 1	64	Hidden 1	96
	Hidden 2	32	Hidden 2	96
	Hidden 3	32	Hidden 3	96
	Hidden 4	32	Hidden 4	96
	Output	1	Output	1
	Activation	*Sigmoid*	**Activation**	*ReLU*
B01P01	Input	96	Input	96
	Hidden 1	64	Hidden 1	96
	Hidden 2	64	Hidden 2	64
	Hidden 3	32	Hidden 3	32
	Hidden 4	32	Hidden 4	64
	Hidden 5	128	Hidden 5	96
	Hidden 6	64	Hidden 6	64
	Hidden 7	96	Hidden 7	64
	Hidden 8	64	Hidden 8	64
	Hidden 9	64	Hidden 9	128

Continued on next page

Table 16: Table shows the architecture for each neural network of class 0 (*no precipitation*) and class 1 (*precipitation*), for each location. There is a clear difference of how many layers is needed for the two location to solve the task. (Continued)

Architecture of the network			
Class 0		**Class 1**	
Layer	**Units**	**Layer**	**Units**
Output	1	Output	1
Activation	*ReLU*	**Activation**	*ReLu*

13.3 Forecast verification

Table 17 show the mean absolute error (MAE), bias and skill score (SS, skill compared to guessing $0.0\,mmh^{-1}$ for every time step) for both the total forecast (i.e containing predictions from both classes) and the forecast for the individual classes.

The skill score for class 0 (*no precipitation*) for both locations is zero, meaning there is no increase or decrease in skill compared to simply guessing zero precipitation for every time-step. This is indicative of the neural networks simply predicting $0\,mmh^{-1}$ precipitation for all samples of class 0, and this was confirmed when plotting the class 0 predictions for both locations. In other words, the neural networks for class 0 is unable to identify any precipitation-events within the data. This is an interesting results and can be attributed to the fact that there is simply not enough precipitation samples within the training data of class 0 for the neural networks to sufficiently learn and identify these events. The percentage of precipitation-events within the training data for class 0 is 12.6% and 8.1% for Florida and Blindern respectively, and this is clearly not enough within the relatively short time-period of January 2020 until August 2022. This raises the question whether it would be sufficient to stop after the classification of class 0 using the random forest classifier (RFC), and only bring class 1 along to the subsequent step of the regression task using neural networks. This is of course not ideal considering that there is in fact precipitation at around 10% of the samples in class 0. However, with this limited amount of precipitation samples within class 0, it would have been sufficient in this thesis as there was no actual gain in training neural networks for this class at either location. One last point regarding this is that a deeper network for class 0 of F01P01 could possibly work, considering there is 12.6% precipitation, but this is speculative.
It is worth noting that the neural network's mean absolute error of class 0 for Florida is 17% smaller than MEPS, and 19% smaller for Blindern, despite the NNs only predicting $0.0\,mmh^{-1}$ for every time step of the class. This emphasizes how important it is to look at all the verification metrics to get a more accurate picture of the neural networks performance.

The verification metrics for the total forecast (combined predictions from both class 0 and class 1) of both locations are improved compared to MEPS with both a *decrease* in mean absolute error (MAE) and an *increase* in skill score (SS). For the total forecast of Florida, the neural network (NN) got a MAE of 0.178, resulting in a 20% *decrease* compared to MEPS (0.223). In addition, the bias for the NN is more than halved compared to MEPS, with 0.027 and 0.061 for the NN and MEPS respectively. While the NN achieves a skill score of 0.99, MEPS got a negative skill score, -0.26, indicating a higher MAE compared to guessing $0\,mm$ precipitation for all samples. The neural network for Blindern has a decrease of 32% of the MAE compared to MEPS (0.065 and 0.097 respectively), accompanied with a skill score of 0.034. Similarly to Florida, MEPS has a negative skill score with -0.433 for Blindern. However, while the bias for the Florida NN is significantly decreased compared to MEPS, for Blindern MEPS has a smaller bias than the NN (in absolute terms).
Looking specifically at the verification metrics for class 1 of Florida, we see an even greater

decrease in MAE for the NN compared to MEPS (32%), but this is accompanied with an negative bias, which is greater than for MEPS in absolute terms. There seem to be a pattern of an overall improvement in both the mean absolute error and the skill score for the neural networks compared to MEPS, while on the other hand they have a varying bias.

It is rather surprising that MEPS achieve a negative skill score across all forecasts (note that the skill is in reference to guessing 0 mmh^{-1} at all time-steps). Looking at the plot of the entire test period in fig. D.11 and D.12 in App. D.3.1, it is quite clear that MEPS does a rather good job at following the general variations of the observed precipitation. Especially for Blindern, it is evident that MEPS in fact does a better job than the neural network at estimating the precipitation amount. This is not what is reflected in the verification metrics discussed above, where the neural network has a 32% smaller mean absolute error than MEPS, as well as a positive skill score. Though these metrics show quite a significant improvement, the plot in fig. D.12 quickly illustrate a problem of verification. The metrics might point to the neural network as the best fit to the observation, but human intuition and pattern recognition can quite easily spot that MEPS in fact seems to be the better fit. This highlight the fact that what the forecasters subjective impression of which is the better forecast is not necessarily coinciding with the verification measures.

Table 17: Mean absolute error, bias and skill score for F01P01 and B01P01, evaluated on the *test* data spanning the period 18th February 2022 16:00 UTC until 3rd August 12:00. *Total* refers to the final product of the class 0 forecast and class 1 forecast combined, while the *class 0/1* refers to the individual classes (*no precipitation/precipitation*). The verification metrics is calculated for all rain rates, i.e all ranges of $RR \geq 0.1$.

			MAE	**Bias**	**SS**
F01P01	**Total**	Neural network	0.178	0.027	0.099
		MEPS	0.223	0.061	-0.126
	Class 0	Neural network	0.047		0.0
		MEPS	0.056	-0.029	-0.201
	Class 1	Neural network	0.602	-0.306	0.12
		MEPS	0.759	0.054	-0.109
B01P01	**Total**	Neural network	0.065	-0.05	0.034
		MEPS	0.097	0.009	-0.433
	Class 0	Neural network	0.039	-0.039	0.0
		MEPS	0.048	-0.023	-0.243
	Class 1	Neural network	0.56	-0.265	0.075
		MEPS	1.004	0.599	-0.66

As mentioned, tab. 17 show that despite the mean absolute error and skill score is improved for the neural networks, the bias is varying, and the plot in fig. D.12 indicate systematic underestimation of precipitation amount. In order to investigate this further, it is necessary to categorize the hourly accumulated precipitation amount into *hourly rain rates, RR* and look at the neural network's performances within these separate rain rates. Looking at the verification metrics for these different rain rates in tab. 18, there is a general pattern of the neural networks performing

better than MEPS for no rain ($RR < 0.1\,mmh^{-1}$) and light rain ($RR \in [0.1, 2.5\rangle\,mmh^{-1}$) for both location. The skill score for Florida within the light rain range is more than two folds higher than MEPS. The exception is the bias, which is for light rain better for MEPS, while the bias for no rain is better for the neural networks. For $RR \geq 2.5\,mmh^{-1}$ both the neural network and MEPS has an increasingly worse performance with increasing rain rate, but MEPS has a general better performance than the neural networks. This is especially evident for Blindern, where MEPS has a skill score between 2 and 1.5 times *higher* for moderate and heavy rain respectively. This is also confirmed when looking at the plot of the entire forecast period, most notably for Blindern (fig. D.12 in App. D.3.1), where an artificial cut-off is very evident. This is similar to the one found from the 1-layer neural network for Florida previously discussed in sec. 13.2 (i.e fig. D.10 in App. D.3). Florida has more realistic variability in the forecast (see fig. D.11) and seem to follow MEPS quite closely. However, during the summer months, there seem to be a underestimation of the amount. While MEPS show similar pattern during the same months, it also overestimates the precipitation amount, and thus don't have the same prevalence of underestimating.

Table 18: Mean absolute error (MAE), bias and skill score (SS) for the *total* forecast for Florida (F01P01, upper) and Blindern (B01P01, lower). The verification metrics are here seperated into the different ranges of hourly *rain rate*, where *no rain* equals $RR < 0.1\,mmh^{-1}$, *light rain* equals $RR \in [0.1, 2.5\rangle\,mmh^{-1}$, *moderate rain* equals $RR \in [2.5, 7.5\rangle\,mmh^{-1}$ and *heavy rain* equals $RR \geq 7.5\,mmh^{-1}$. *All rates* is equvialent ot the values presented in tab. 17 for the *total* forecast, and is included to ease comparison.

			All rates	**No rain**	**Light rain**	**Moderate rain**	**Heavy rain**
F01P01	**MAE**	NN	0.205	0.042	0.587	2.768	7.986
		MEPS	0.223	0.061	0.627	2.553	7.349
	Bias	NN	-0.068	0.042	-0.231	-2.723	-7.986
		MEPS	-0.009	0.061	-0.042	-2.324	-7.349
	SS	NN	-0.035	-inf	0.067	0.255	0.08
		MEPS	-0.126	-inf	0.002	0.313	0.153
B01P01	**MAE**	NN	0.065	0.006	0.493	3.339	10.547
		MEPS	0.097	0.035	0.609	2.745	10.382
	Bias	NN	-0.05	0.006	-0.44	-3.339	-10.547
		MEPS	0.009	0.035	-0.062	-2.481	-9.79
	SS	NN	0.034	-inf	0.148	0.078	0.01
		MEPS	-0.433	-inf	-0.052	0.242	0.025

Because tab. 18 show a notable difference in the neural networks' performance with increasing hourly rain rates (RR), the distribution of the forecasts and observations was plotted for each location, as seen in fig. 22. Looking at the distribution of light rain for Florida (the upper panel of fig. 22a), it is clear the the neural network follow MEPS quite closely for the most part, and they are in an overall good agreement with the observations (with the exception of very low rain rate). Looking at $RR \in [2.5, 5\rangle\,mmh^{-1}$ (middle panel) there is a greater difference between the distribution of both MEPS and the neural network compared to the observations, where the latter usually has a higher density for most rain rates within this given range. It also becomes

clear that the neural network has a significant lower density than MEPS, especially in the range $RR \in [2.5, 4\rangle\, mmh^{-1}$. In the range of $RR \in [5, 7.5\rangle\, mmh^{-1}$ there are few observations and very low density. In this range there are no predictions from the neural network, as suspected.
The distribution for Blindern is shown in fig. 22b. Initially, the distribution for light rain (upper panel) looks similar to what was found for Florida. However, for $RR \in [1.25, 2.5\rangle\, mmh^{-1}$ the neural network has completely disappeared, meaning there is no predictions from the neural network in this range. The higher rain rates in the middle and lower panel confirm that this is the case for *all* $RR \geq 1.25\, mmh^{-1}$, showing a quite severe and systematic underprediction of the precipitation amount.

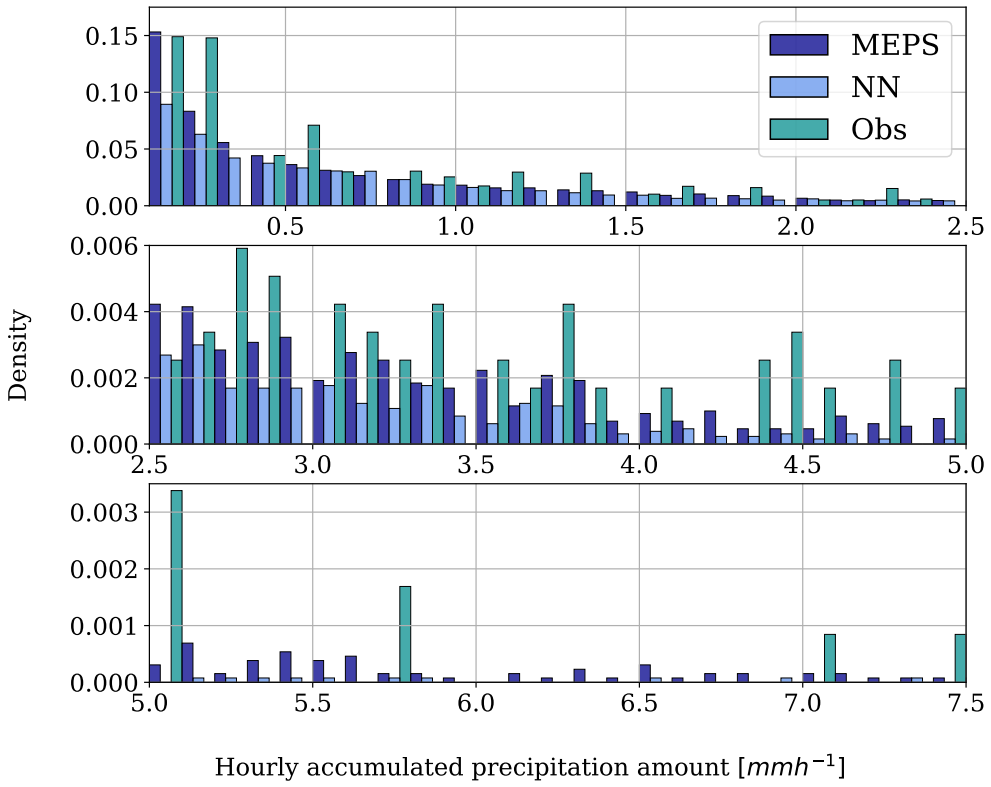
To illustrate how the neural networks performs in both winter and summer, an arbitrary forecast from February and July was selected for each location. Figure 23 shows the winter and summer forecast for Florida (upper panels) and the difference between forecast and observation ($forecast - observation$, lower panels). Figure 23a show the 18UTC forecast from 17 February 2022, and show that MEPS clearly overestimates the precipitation amount in the beginning of the forecast period, while both MEPS and the neural network underestimates the precipitation amount of the event at 50 hours forecast lead time. In general though, it seems like both MEPS and the neural network is mostly able to identify the significant precipitation-events, and is very much able to identify the no precipitation event in the middle of the forecast period. From the summer months the 00UTC forecast from 3 July 2022 is plotted in 23b. Both MEPS and the neural network are able to identify most precipitation-events, but struggle to both identify and predict the major precipitation-event with heavy rain rate that is observed around 8 hours forecast lead time.

Looking at the lower panels of fig. 23, its seems the neural network follow MEPS quite closely for the most part, with the exception of the overestimation by MEPS at approximately 5 hours forecast lead time in fig. 23a. The fact that the neural network follow MEPS closely is not surprising as MEPS' hourly precipitation forecast is not only one of the predictands used by the neural network, it also has the highest Spearman rank.

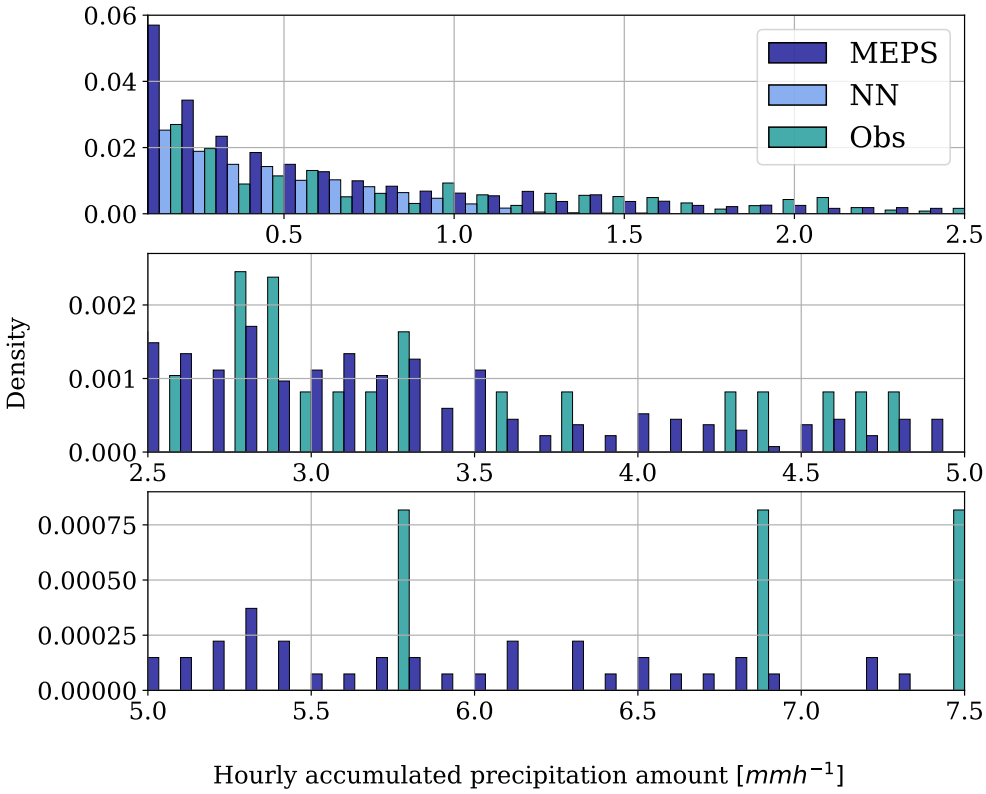
Figure 24 shows the equivalent plot for Blindern. Looking at the 6UTC forecast from 19 February 2022 in fig. 24a it is quite clear that MEPS overestimates the occurrence of precipitation-events. While the neural network seems to predict number of occurrences closer to the number of observed, it seems at times to predict the precipitation one or two hours earlier than it is observed. It should be noted that though the neural network might be slightly off hour-to-hour, this is still quite good. The neural network's predictions also looks reasonable in consideration of precipitation amount.

The 00UTC forecast from 1 July 2022 is shown in fig. 24b. Both of the models struggle to identify the precipitation-event at 14 hours forecast lead time, however, they both seem to identify the no precipitation-events rather well. Looking at the precipitation-event at 29 hours forecast lead time, it seems that both MEPS and the neural network is a few hours early with the predicted precipitation. In addition, the neural network's underestimation of precipitation amount is very clear in this instance. MEPS seems to be able to predict more realistic precipitation amount than the neural network in this particular forecast.

These are simply a few selected forecast presented to illustrate the performance of the neural networks compared to MEPS and the observations, both during winter months and summer months. Keep in mind that this is an hour-to-hour comparison, but the neural networks show much of the same variability as the observation, though more successfully for Florida than Blindern.



(a)



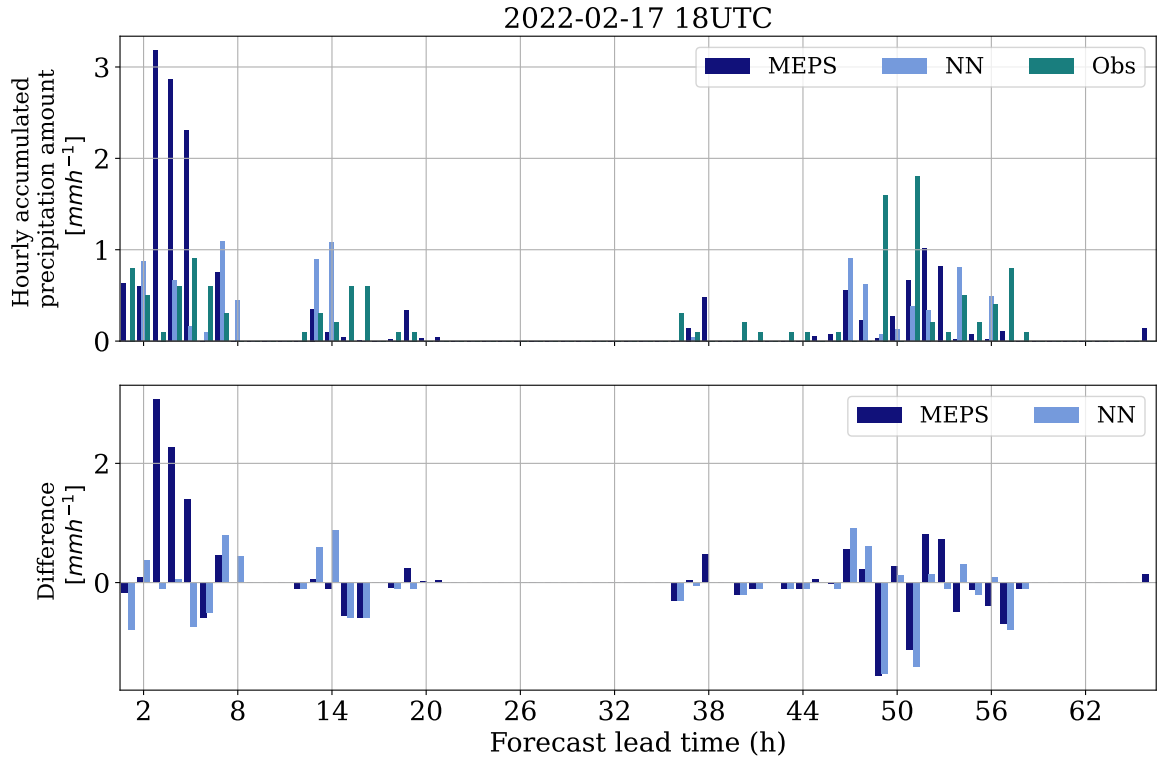
(b)

Figure 22: An overview of the distribution of the final forecast for (22b) Florida (F01P01) and (22b) Blindern (B01P01). The figure shows the distribution for both the neural network (light blue), MEPS (navy) and the observations (teal). Note that both the x-axis and the y-axis changes between the subplots in order to accurately show the distribution of the different rain rates.

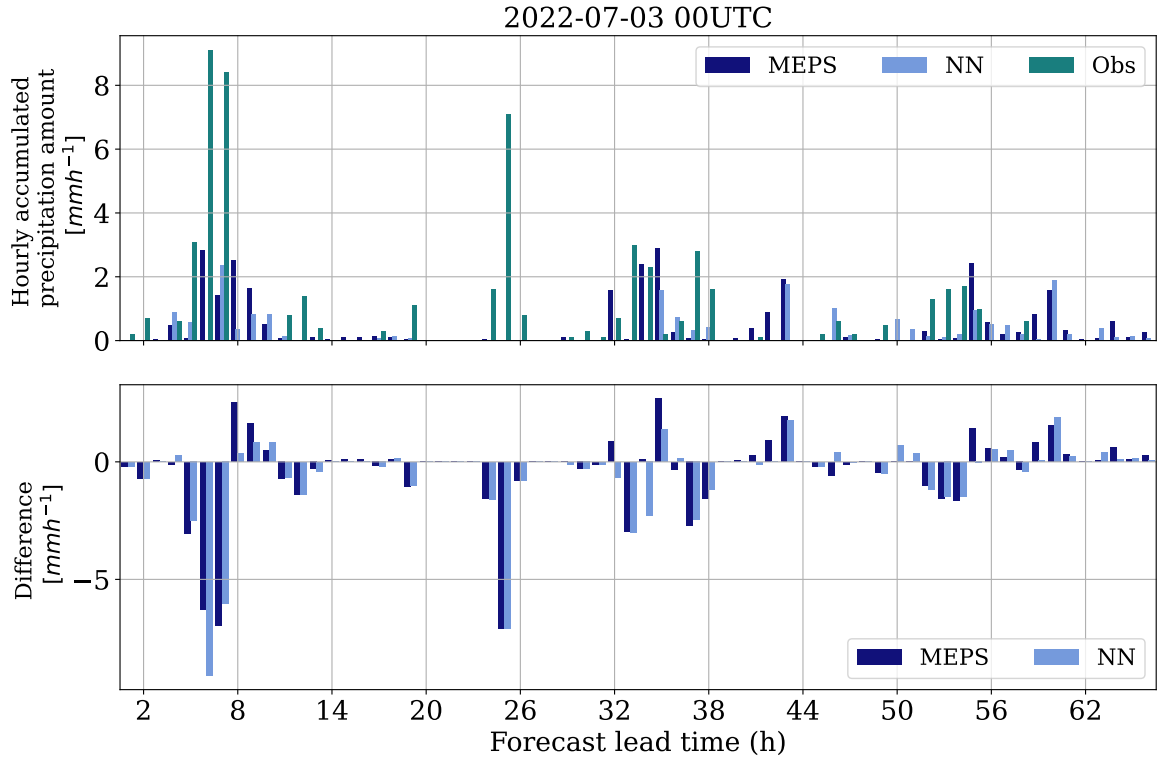
Though the neural network performs quite well for light rain ($RR \in [0.1, 2.5\rangle\, mmh^{-1}$) due to the majority of precipitation samples being classified as such, they are unable to capture moderate to heavy rain rates. Capturing the full variability of precipitation seems to be a difficult task for both the neural networks and to some degree MEPS, though it is clear that MEPS is able to do so at a higher degree than the neural networks. The neural networks for Blindern had a 67% and 60% *decrease* in skill for moderate and heavy rain compared to MEPS, while Florida had a 19% and 48% *decrease*. Once again the lack of data, in this case the small amount of precipitation samples in the range of the heavier rain rates, is evident in the neural network as a systematic underprediction. It seems quite clear that the amount of data is mostly to blame for this fault, as it is even more severe for Blindern than Florida, which has only 12% precipitation versus 32%.

The forecast verification, both for the neural networks and MEPS, is assessed hour-to-hour and point-to-point. This means that any spatial or temporal shifts is not taken into consideration, i.e if the models forecast a precipitation field and the spatial structures are quite close to the observed, but slightly skewed in some direction, the traditional verification metrics will not accurately reflect the true characteristics of the forecast [Gilleland et al., 2009].

The neural networks show good improvement (relative to MEPS) according to the verification metrics. Upon further inspection, it becomes evident that the good performance is limited to hourly rain rates of less than $2.5\, mmh^{-1}$, and that MEPS outperform the neural networks for all rates above this threshold. Looking at fig. D.11b and D.12b in App. D.3.1 it seems that the neural networks follow the same overall pattern as MEPS quite closely, which is not surprising considering MEPS *hourly precipitation* forecast is the most important feature. Looking at specific forecasts however, figures 23 and 24 confirm that there are differences between the predicted values from the neural networks and MEPS, showing that the former does not simply reproduce the latter model's predictions directly, confirming that learning has been achieved.
The neural networks has an overall good performance, and even though there is a notable decrease in mean absolute error and an increase in skill relative to MEPS, the distribution plotted in fig. 22 makes it clear that one cannot conclude that the NN are particularly better than MEPS altogether. However, at some areas, in particular for *no rain* and *light rain* the neural network perform quite well and is at the very least comparable to MEPS (for Florida), which in itself is a notable achievement and very much a promising result for the future of machine learning in numerical weather prediction.

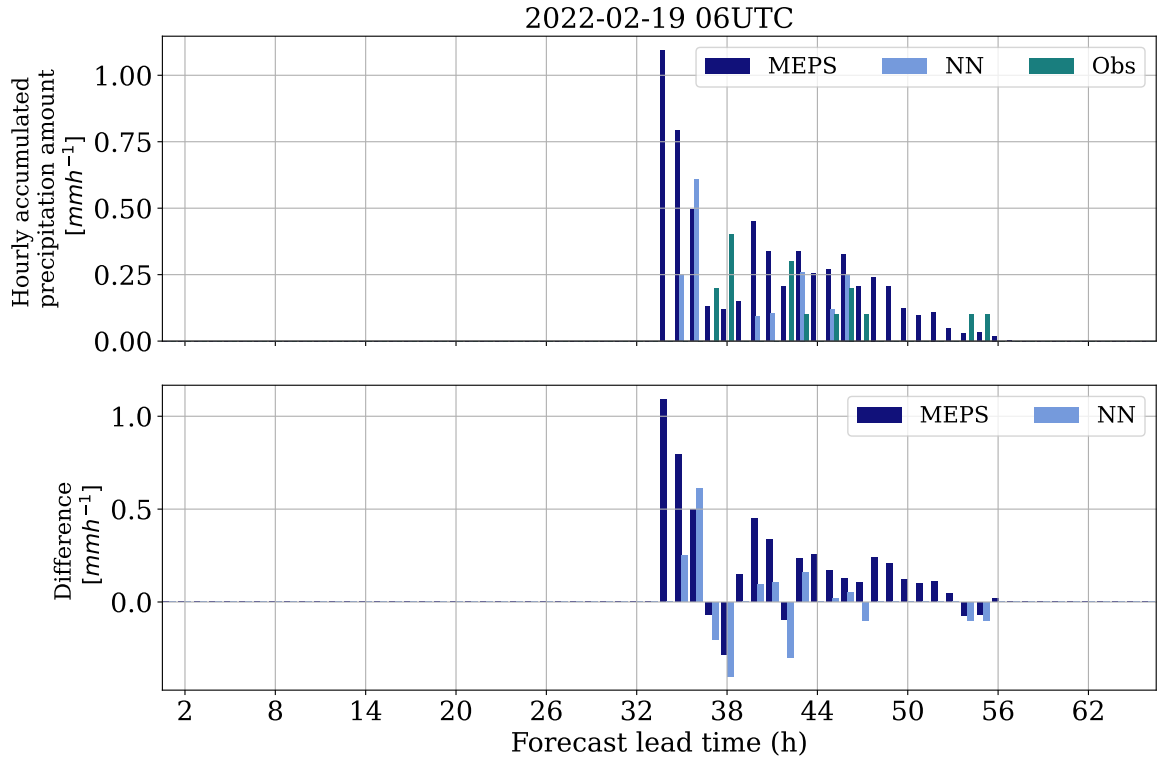


(a)

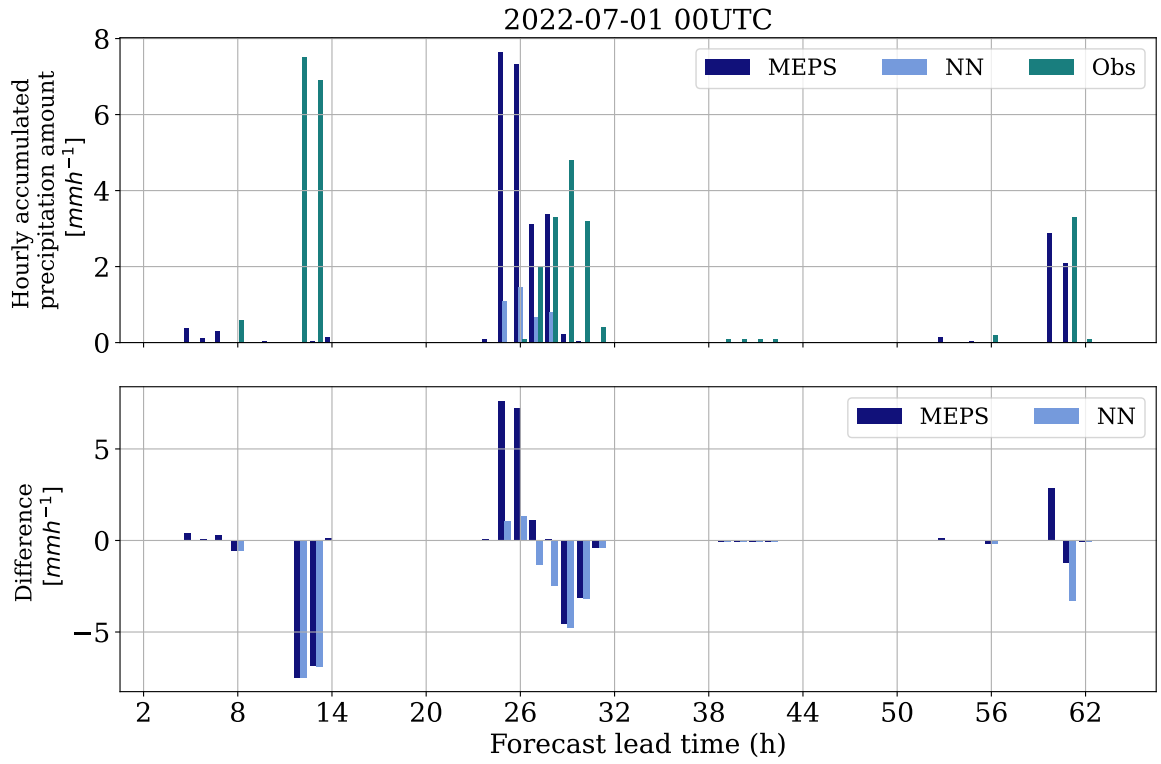


(b)

Figure 23: Precipitation-events for two specific forecasts, and the accompanying observations. Navy bar shows MEPS forecast, light blue shows the neural network forecast and teal shows the measured precipitation. Fig. (a) shows a particular forecast from the winter season, more specifically the 2022/02/17 18 UTC forecast, while (b) shows a forecast from the summer months, 2022/07/03 00 UTC forecast.



(a)



(b)

Figure 24: Same as figure 23, but for Blindern (B01P01). Navy bar shows MEPS forecast, light blue shows the neural network forecast and teal shows the measured precipitation. Fig. (a) shows a particular forecast from the winter season, more specifically the 2022/02/17 18 UTC forecast, while (b) shows a forecast from the summer months, 2022/07/03 00 UTC forecast.

Part V
Conclusion

14 Conclusions

The method of splitting the data into training, validation and test data proved to be very important. It was first decided to split the data randomly to get an even distribution of precipitation samples across the different datasets. Because of the limited time-period in the data (2020-2021), the precipitation distribution when splitting chronologically was varying to a significant degree across the different datasets. However, after all the steps in this thesis was done, including feature selection, training a random forest classifier and training a neural network for each class assigned by the classifier, a final evaluation on 2022 data showed significantly poorer performance than the evaluation done on the randomly drawn test data, going from a 30% *increase* in skill to a 8.3% *decrease* in skill for class 1 (*precipitation*) compared to MEPS. This illuminated the effect of the auto-correlation within the data leading to overly optimistic estimates of model performance, just as Schultz et al. [2021] emphasized, to such a degree that it was deemed necessary to redo the random forest classifier and neural network process with chronologically split data.

Even though the feature selection was done on randomly split data, it is not likely that the outcome would differ if done on chronologically split data due to the nature of the features selected. There was similar features selected for both location, such as *low* and *medium cloud cover, relative humidity* and *hourly precipitation* (MEPS precipitation forecast). Location-specific features proved to be *zonal wind* for Florida and *meridional wind* for Blindern, as expected due to the topography and climate of the locations. The features selected was assigned high importance by the random forest classifier, as well as having a high Spearman rank, further supporting the choice of selected features. However, many similar features was selected, such as several cloud cover features, therefore it is likely that the set of features could be simplified even further. In addition, a bigger emphasize on feature engineering could prove useful both in potentially further eliminate any redundant information across the selected features, and translating information into a form which is easier to interpret for the machine learning algorithms. A separate feature selection specifically aimed at the neural networks would also be recommended in further work.

The objective used by the grid-search performed by GridSearchCV for hyperparameter tuning does not reflect the metrics traditionally used in forecast verification, and thus the grid-search done in this thesis did not select the optimal random forest classifier (RFC) for the task. The RFC selected by the grid-search got a skill score of 0.42 and -0.37 For Florida and Blindern respectively, while the *base* RFC scored 0.49 and 0.28 in comparison. It could be favorable to manually do a grid-search using the verification metric appropriate for such a task, for example skill score, but due to time-limitations this was not executed in this thesis. Considering the performance of the RFC proved to be quite influenced by the hyperparameters, one can easily imagine even better results if a proper grid-search (that actually provide the optimal model) was performed.
The random forest classifiers for Florida are able to identify more precipitation-events than MEPS, whilst they struggle more for Blindern, though this is not reflected in the skill score.

For the training of the neural networks the performance benefited somewhat from incorporating the tendency features in the training data as it introduced some temporal dependencies across one time-step and the prior. Including the tendency features resulted in a *decrease* in

mean absolute error (MAE) between $5-8\%$ for the validation data, however, a separate feature selection for neural networks would be advisable in further work.
Not so surprisingly the architecture of the neural networks proved to be of great significance to the overall performance. Going from one hidden layer to four (F01P01) removed the artificial cut-off of the precipitation amount seen from the former model's predictions (fig. D.10 and D.11a), making the latter model's predictions more realistic and *reducing* the MAE of the moderate and heavy rain rates with 11% and 3% respectively. Though the latter is somewhat modest, the skill score (SS) went from 0.168 to 0.255 for moderate rain, and 0.08 to 0.057 for heavy rain (equivalent to a 51% and 40% *increase*). The optimal neural networks for Blindern needed nine hidden layers to solve the regression task, yet still did not manage to capture the variability of hourly precipitation amount and had a similar artificial cut-off as seen for the 1-layer neural network for Florida.

The neural network has a general reduction in mean absolute error (MAE) and increased skill score (SS) compared to MEPS. The decrease in MAE for the Florida NN is found to be 20% for the total forecast and 32% for class 1 (*precipitation*) specifically, while the equivalent for Blindern is 32% and 45% respectively. However, a general pattern is that the neural networks often underestimate the precipitation amount, and from fig. 22 in sec. 13.2 it is clear that the neural networks struggle to capture the heavier rain rates. In comparison to MEPS, the neural network got a 19% and 48% *decrease* in skill for Florida (F01P01) in the moderate and heavy rain rates ($RR \geq 2.5\, mmh^{-1}$), and a even more severe reduction for Blindern (B01P01) with a 67% and 60% *decrease* in skill. MEPS also struggles more with the heavier rain rates, however, it is clear it outperforms the neural networks in these ranges. This is likely due to the very limited samples within these ranges, in other words there is simply too few samples in these ranges for the neural networks to learn sufficiently. The verification metrics show good results for the total forecast of the neural networks and this is mainly because most of the samples fall within no rain ($RR < 0.1\, mmh^{-1}$) and light rain ($RR \in [0.1, 2.5\rangle\, mmh^{-1}$) range, and thus the good performance within these ranges hides the problem in the moderate and heavy rain rates of $RR \geq 2.5\, mmh^{-1}$. Consequently, it is clear that the verification metrics does not give the full picture, and the comparison between the models are challenging to accurately depict.

As mentioned, the neural networks do quite well in the range of hourly rain rate $(RR) \in [0.1, 2.5\rangle$, particularly for Florida, and this is undoubtedly because the majority of precipitation data fall within this range, thus it is this range the machine learning algorithm is able to learn well. The neural networks for class 0 (*no precipitation*) at both locations was not able to identify and predict any precipitation-events within the class, which makes up about 12% for Florida and 8% for Blindern. This is simply not enough precipitation-samples to solve the task. Therefore, in this thesis it would be sufficient to predict $0\, mmh^{-1}$ for all samples classified as class 0 by the random forest, and not train a neural network for this particular class.

Splitting the dataset depending on forecast lead time (i.e P02-6h, P02-12h, P02-66h) did not improve model performance enough to justify the extra steps such a split would require operational use. In fact, for Blindern it actually resulted in less increase in skill relative to MEPS, than what was seen for P01. Furthermore, there was some difference in model performance of the random forest classifiers when trained on 10 versus 30 features (approximately 10% increase in skill for 30 features). This increase was however not proportional to the increase in features, and this emphasize the redundant information found within the dataset. Simplicity is favorable to avoid overfitting and consequently, only F01P01 and B01P01, referring to the datasets with variables directly from MEPS and the entire forecast lead time (i.e 66 hours), was used for all three steps of the thesis. The fact that fewer features and one single model for the full forecast period proved sufficient further support that simplicity is a favorable starting point for solving the task.

Machine learning thrives on big data, and this has been highlighted at each step of this thesis. Though there are a decent number of samples (training data contain 134 560 samples), the problem is that the samples that actually classifies as precipitation is very much a minority. Florida has 30% precipitation samples in the training data, while Blindern has only 14.6%, and this difference has significantly influenced the performance of both the random forest classifier and the neural networks.

In this thesis, the full range of forecast lead time (66h) is used and then paired up with the observation corresponding to the given time-step. In practice, this means that there are several samples for each specific time step, ranging from early to late in the forecast (early referring to few hours forecast lead time, while late refers to a significant number of hours forecast lead time), which are all paired with the same observation. In other words, a forecast at 6 hours forecast lead time could be paired with the same observation as a forecast at 60 hours forecast lead time (given they both forecast the same time-step). Due to the chaotic nature of the atmosphere, the forecast at 6 hours forecast lead time is in most cases closer to the actual state of the atmosphere at said time-step (compared to 60 hours forecast lead time), and thus gives a more realistic representation of the physics resulting in the specific observation. This raises the question whether the machine learning algorithm could be confused by the many samples, especially those with a higher forecast lead time, and whether it would be better to only use data from shorter forecast lead times in the training of the machine learning algorithms, while still using the final model on all forecast lead times. This would of course reduce the number of samples drastically, which would have a negative effect on the machine learning algorithms' ability to generalize, so it is uncertain whether the potential gain from limiting the forecast lead time in training would be greater than the known loss from the far fewer data-points this limitation would result in. The skill of the random forest classifiers of P02-6h and P02-12h (tab. C.15 in App. C.1.3) is comparative to the skill of P01 despite far less samples, though the *increase* in skill (compared to meps) is lower for the shorter lead times with fewer samples. However, for the neural networks in this thesis this trade-off would most certainly result in a greater loss due to the limited time-period the data is collected from. In instances were many years of data is available though, this could potentially provide a significant gain.

The verification methods used is in this thesis is a direct comparison hour-by-hour, which does not take into account temporal shifts between forecast and observations, in other words, whether the model made good predictions, but an hour early or late, which would matter in reality. Another important thing to note is that MEPS is able to forecast precipitation pretty well, and provide good quality predictands used by the neural network. The difference between the spatial and temporal resolution of the model output and the observations used during verification is called the representativness error [Warner, 2011]. In this thesis the temporal resolution of the model output and the observations is both hourly, however, the spatial resolution of the model grid i $2.5km$, while the observations is from only one station situated within the grid.

15 Further work

Lastly, this section will summarize aspects of this thesis that would be interesting to look further into.

One of the clear implications during this work has been the unbalance between precipitation and no precipitation samples within the data. In addition, the unbalance has varied between the training, validation and test data has also been a challenge. Increasing the time-period would

mostly even out the differences between the datasets as they would span years instead of a few season. This would not solve the general low percentage of precipitation though. To address this, over- or under-sampling could be used to get the percentage up to a sufficient level so the machine learning algorithm had enough samples to learn from. A longer time-period would also result in more samples, not just overall, but also in the heavier rain rates, which could further improve the neural networks ability to capture these ranges better.

Another aspect of interest is additional feature selection and engineering. It would be interesting to do further work with the feature selection and engineering in an effort to remove redundant information and possibly translate information to a form easier interpreted by the machine learning algorithms. Particularly wind normal to topography could be a variable of interest, as this could could make it possible to combine Florida and Blindern to one big dataset, without the machine learning algorithm learning a systematic bias towards Florida-specific features, such as zonal wind (due to far more precipitation data for Florida compared to Blindern).

Furthermore, a manual grid-search to select the optimal hyperparameters for the random forest classifiers, as well as an even more thorough grid-search to select the architecture of the neural network, looking into other types of layers and perhaps try other types of networks such a recurrent neural network or a long short-term memory neural networks to capture the temporal dependencies within the domain. This would require a significant change of the structure of the data, as it now is compiled of many small time-series (i.e one forecast of 66 hours is one continuous time-series).

As described in the last paragraph in the section above, it would be interesting to look into whether using MEPS data from shorter forecast lead times makes better training of the neural network. Another option would be to categorize the precipitation based on hourly rain rate (i.e light rain, moderate rain, heavy rain), and instead of trying to predict the actual amount the task would be to predict rain rate category. This reduces the problem to a multi-class classification task. This would likely only be sufficient for recreational use, and even then the higher rain rates cover a too wide range. However, it would still be valuable information, because it can be argued that it is not the exact precipitation amount that is of importance for most people, but how heavy the rainfall will be (i.e rain rate).

References

David Ahijevych, James O. Pinto, John K. Williams, and Matthias Steiner. Probabilistic forecasts of mesoscale convective system initiation using the random forest data mining technique. *Weather and Forecasting*, 31:581–599, 2016. ISSN 15200434. doi: 10.1175/WAF-D-15-0113.1.

C. Donald Ahrens and Robert Henson. *Essentials of meteorology: An invitation to the Atmosphere*. Cengage Learning, 8th edition, 2018. ISBN 9781305628458.

Peter Bauer, Alan Thorpe, and Gilbert Brunet. The quiet revolution of numerical weather prediction. *Nature*, 525:47–55, 2015. ISSN 14764687. doi: 10.1038/nature14956.

Lisa Bengtsson, Ulf Andrae, Trygve Aspelien, Yurii Batrak, Javier Calvo, Wim de Rooy, Emily Gleeson, Bent Hansen-Sass, Mariken Homleid, Mariano Hortal, Karl Ivar Ivarsson, Geert Lenderink, Sami Niemelä, Kristian Pagh Nielsen, Jeanette Onvlee, Laura Rontu, Patrick Samuelsson, Daniel Santos Muñoz, Alvaro Subias, Sander Tijm, Velle Toll, Xiaohua Yang, and Morten Ødegaard Køltzow. The harmonie-arome model configuration in the aladin-hirlam nwp system. *Monthly Weather Review*, 145:1919–1935, 5 2017. ISSN 15200493. doi: 10.1175/MWR-D-16-0417.1.

Vilhelm Bjerknes. Das problem der wettervorhersage, betrachtet vom standpunkte der mechanik und der physik (the problem of weather prediction, considered from the viewpoints of mechanics and physics) (translated by esther volken and stefan brönnimann. -meterol z. 18 (2009), 663-667). *Meteorologische Zeitschrift*, 21, 1904. ISSN 09412948. doi: 10.1127/0941-2948/2009/416. (Translated by Esther Volken and Stefan Brönnimann. -Meterol Z. 18 (2009), 663-667).

Leo Breiman. Random forests. 45:5–32, 2001.

Leo Breiman, Jerome H. Friedman, Richard A. Olshen, and Charles J. Stone. *Classification and regression trees*. 1984. ISBN 0534980538.

The Editors of Encyclopaedia Britannica. Orographic precipitation. URL `https://www.britannica.com/science/orographic-precipitation`.

Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122, 2013.

J. G. Charney, R. Fjørtoft, and J. von Neumann. Numerical integration of the barotropic vorticity equation. *Tellus*, 2:237–254, 11 1950. ISSN 00402826. doi: 10.1111/j.2153-3490.1950.tb00336.x.

Petter Dannevig. Hordaland - klima i store norske leksikon, a. URL `https://snl.no/Hordaland_-_klima`.

Petter Dannevig. Oslo - klima i store norske leksikon, b. URL `https://snl.no/Oslo_-_klima`.

Inger Lise Frogner, Andrew T. Singleton, Morten Køltzow, and Ulf Andrae. Convection-permitting ensembles: Challenges related to their design and use. *Quarterly Journal of the Royal Meteorological Society*, 145:90–106, 9 2019. ISSN 1477870X. doi: 10.1002/qj.3525.

David John Gagne, Amy McGovern, Sue Ellen Haupt, Ryan A. Sobash, John K. Williams, and Ming Xue. Storm-based probabilistic hail forecasting with machine learning applied to convection-allowing ensembles. *Weather and Forecasting*, 32:1819–1840, 2017. ISSN 15200434. doi: 10.1175/WAF-D-17-0010.1.

Eric Gilleland, David Ahijevych, Barbara G. Brown, Barbara Casati, and Elizabeth E. Ebert. Intercomparison of spatial forecast verification methods. *Weather and Forecasting*, 24:1416–1430, 10 2009. ISSN 08828156. doi: 10.1175/2009WAF2222269.1.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Deep learning, 2016.

Gregory Hakim and Jérôme Patoux. *Weather: A concise introduction*. Cambridge Univeristy Press, 2018. ISBN 9781108417167.

Inger Hanssen-Bauer and Eirik J. Førland. Annual and seasonal precipitation variations in norway 1896-1997. 1998.

Inger Hanssen-Bauer, Ole Einar Tveito, Helga Therese, Tilley Tajet, and Reidun Gangstø Skaland. Temperatur-og nedbør-regioner i norge sammenligning av forskjellige regioninndelinger. 2022. ISSN 2387-4201. URL `www.met.no`.

Rohan Joseph. Learn with an example : Hierarchical clustering, 2018. URL `https://medium.com/@rohanjoseph_91119/learn-with-an-example-hierarchical-clustering-873b5b50890c`.

Eugenia Kalney. *Atmospheric modeling, data assimilation and predictability*. Cambridge Univeristy Press, 2006. ISBN 9780521796293.

Ryuji Kimura. Numerical weather prediction, 2002.

Norsk Klimaservicesenter. Seklima observasjoner og værstatistikk. URL `https://seklima.met.no/`.

Max Kuhn and Kjell Johnson. *Applied Predictive Modeling*. Springer New York, NY, 1 edition, 2003. ISBN 978-1-4614-6849-3. doi: https://doi.org/10.1007/978-1-4614-6849-3.

Yiwen Mao and Asgeir Sorteberg. Improving radar-based precipitation nowcasts with machine learning using an approach based on random forest. *Weather and Forecasting*, 35:2461–2478, 12 2020. ISSN 15200434. doi: 10.1175/WAF-D-20-0080.1.

Ryan M. May, Sean C. Arms, Patrick Marsh, Eric Bruning, John R. Leeman, Kevin Goebbert, Jonathan E. Thielen, Zachary S Bruick, and M. Drew. Camron. Metpy: A Python package for meteorological data, 2022. URL `https://github.com/Unidata/MetPy`.

Amy McGovern, Kimberly L. Elmore, David John Gagne, Sue Ellen Haupt, Christopher D. Karstens, Ryan Lagerquist, Travis Smith, and John K. Williams. Using artificial intelligence to improve real-time decision-making for high-impact weather. *Bulletin of the American Meteorological Society*, 98:2073–2090, 2017. ISSN 00030007. doi: 10.1175/BAMS-D-16-0123.1.

Tom M. Mitchell. *Machine Learning*. Academic Press, Elsevier, 3rd edition, 1997. ISBN 0070428077.

Allan H. Murphy. What is a good forecast? an essay on the nature of goodness in weather forecasting. 1992.

Malte Müller, Mariken Homleid, Karl Ivar Ivarsson, Morten A.Ø. Køltzow, Magnus Lindskog, Knut Helge Midtbø, Ulf Andrae, Trygve Aspelien, Lars Berggren, Dag Bjørge, Per Dahlgren, Jørn Kristiansen, Roger Randriamampianina, Martin Ridal, and Ole Vignes. Arome-metcoop: A nordic convective-scale operational weather prediction model. *Weather and Forecasting*, 32: 609–627, 2017. ISSN 15200434. doi: 10.1175/WAF-D-16-0099.1.

Tom O'Malley, Elie Bursztein, James Long, François Chollet, Haifeng Jin, Luca Invernizzi, et al. Kerastuner. `https://github.com/keras-team/keras-tuner`, 2019.

J R Quinlan. Induction of decision trees, 1986.

David E. Rumelhart, Geoffrey E. Hilton, and Ronald J. Williams. Learning representation by back-propagating errors. *Nature*, 323:533–536, 1986. doi: https://doi.org/10.1038/323533a0.

Arróliga Araica; Blandón Ruíz, John W. Creswell, Six Byzantine Portraits, Louis Cohen, Lawrence Manion, Keith Morrison, . . , and Bert Creemers. *The Elements of Statistical Learning*, volume 3. 2015. ISBN 0203985982. URL `http://repositorio.unan.edu.ni/2986/1/5624.pdf`.

Arthur L Samuel. Some studies in machine learning. *IBM Journal of Research and Development*, 3:210–229, 1959. URL `https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5392560`.

M. G. Schultz, C. Betancourt, B. Gong, F. Kleinert, M. Langguth, L. H. Leufen, A. Mozaffari, and S. Stadtler. Can deep learning beat numerical weather prediction?, 4 2021. ISSN 1364503X.

Scikit-learn API Reference. Api reference, 2022. URL `https://scikit-learn.org/stable/modules/classes.html#`. 13.07.22.

Scikit-learn User-Guide. User guide, 2022. URL `https://scikit-learn.org/stable/user_guide.html`. 13.07.22.

Shai Shalev-Shwartz and Shai Ben-David. Understanding machine learning: From theory to algorithms, 2014. URL `http://www.cs.huji.ac.il/~shais/UnderstandingMachineLearning`.

Ian Strangeway. *Precipitation*. Cambridge University Press, 2007. ISBN 9780521172929.

Olivier Talagrand. Assimilation of observations, an introduction (gtspecial issueltdata assimilation in meteology and oceanography: Theory and practice). *Journal of the Meteorological Society of Japan. Ser. II*, 75(1B):191–209, 1997. doi: 10.2151/jmsj1965.75.1B_191.

Authors TensorFlow. Keras api. `https://keras.io/api/`, 2015.

Piet Termonia, Claude Fischer, Eric Bazile, François Bouyssel, Radmila Brožková, Pierre Bénard, Bogdan Bochenek, Daan Degrauwe, Mariá Derková, Ryad El Khatib, Rafiq Hamdi, Ján Mašek, Patricia Pottier, Neva Pristov, Yann Seity, Petra Smolíková, Oldřich Španiel, Martina Tudor, Yong Wang, Christoph Wittmann, and Alain Joly. The aladin system and its canonical model configurations arome cy41t1 and alaro cy40t1. *Geoscientific Model Development*, 11:257–281, 1 2018. ISSN 19919603. doi: 10.5194/gmd-11-257-2018.

The Norwegian Meteorological Institute (MET Norway). Meps archive, 2022. URL `https://thredds.met.no/thredds/metno.html`. Accessed: February, March 2022.

John M. Wallace and Peter V. Hobbs. *Atmospheric Science,*. Elsevier, 2nd edition, 2006. ISBN 978-0-12-732951-2. doi: https://doi.org/10.1016/C2009-0-00034-8.

Thomas Tomkins Warner. *Numerical Weather and Climate Prediction*. Cambridge Univeristy Press, 2011. ISBN 9780521513890.

Daniel Wilks. *Statistical methods in the atmospheric sciences*. Academic Press, Elsevier, 3rd edition, 2011. ISBN 9780123850225.

Part VI
Appendix

A Data

A.1 Location

Table A.1: This table shows an overview of the x and y values of the points within a 3.5km radius from Florida (left) and Blindern (right)

Florida		**Blindern**	
x	**y**	**x**	**y**
−532584.	−285017.88	−240084.	−367517.88
−532584.	−282517.88	−240084.	−365017.88
−530084.	−287517.88	−240084.	−362517.88
−530084.	−285017.88	−237584	−367517.88
−530084.	−282517.88	−237584	−365017.88
−527584.	−285017.88	−237584	−362517.88
−527584.	−282517.88	−235084.	−365017.88

A.2 The calculated variables in dataset 02

Table A.2: The additional variables for the 02 dataset, and the equations used in the calculation.

Variable	**Unit**	**Calculation**
MEPS: Label, y	1	eq. 39
Dewpoint temperature at 925hPa	$°C$	eq. 40
Dewpoint temperature at 500hPa	$°C$	eq. 40
Dewpoint temperature at 2m	eq. eq. 40	
Windspeed at 925hPa	ms^{-1}	eq. 41
Windspeed at 850hPa	ms^{-1}	eq. 41
Windspeed at 500hPa	ms^{-1}	eq. 41
Wind direction at 10m	°	eq. 42
Wind direction at 925	°	eq. 42
Wind direction at 850	°	eq. 42
Wind direction at 500	°	eq. 42
Pressure of lifting condensation level	Pa	[May et al., 2022, see metpy.calc.lcl]
Temperature of lifting condensation level	$°C$	[May et al., 2022, see metpy.calc.lcl]
Absolute moisture flux at 925hPa		eq. 43
Zonal moisture flux at 925hPa		eq. 43
Meridional moisture flux at 925hPa		eq. 43
Windspeed tendency 10m	$\frac{ms^{-1}}{hour}$	eq. 44
Windspeed tendency 925hPa	$\frac{ms^{-1}}{hour}$	eq. 44
Windspeed tendency 850hPa	$\frac{ms^{-1}}{hour}$	eq. 44
Windspeed tendency 500hPa	$\frac{ms^{-1}}{hour}$	eq. 44
Air temperature tendency 2m [dT/dt]	$\frac{K}{hour}$	eq. 44

Continued on next page

Table A.2: The additional variables for the 02 dataset, and the equations used in the calculation. (Continued)

Variable	Unit	Calculation
Relative humidity tendency 2m [dRH/dt]	$\frac{1}{hour}$	eq. 44
Zonal wind tendency 10m [du/dt]	$\frac{ms^{-1}}{hour}$	eq. 44
Meridional wind tendency 10m [dv/dt]	$\frac{ms^{-1}}{hour}$	eq. 44
Pressure tendency [dp/dt]	$\frac{Pa}{hour}$	eq. 44
Zonal wind tendency at 925hPa [du/dt]	$\frac{ms^{-1}}{hour}$	eq. 44
Zonal wind tendency at 850hPa [du/dt]	$\frac{ms^{-1}}{hour}$	eq. 44
Zonal wind tendency at 500hPa [du/dt]	$\frac{ms^{-1}}{hour}$	eq. 44
Meridional wind tendency at 925hPa [dv/dt]	$\frac{ms^{-1}}{hour}$	eq. 44
Meridional wind tendency at 850hPa [dv/dt]	$\frac{ms^{-1}}{hour}$	eq. 44
Meridional wind tendency at 500hPa [dv/dt]	$\frac{ms^{-1}}{hour}$	eq. 44
Air temperature tendency at 925hPa [dT/dt]	$\frac{K}{hour}$	eq. 44
Air temperature tendency at 850hPa [dT/dt]	$\frac{K}{hour}$	eq. 44
Air temperature tendency at 500hPa [dT/dt]	$\frac{K}{hour}$	eq. 44
Relative humidity tendency at 925hPa [dRH/dt]	$\frac{1}{hour}$	eq. 44
Relative humidity tendency at 500hPa [dRH/dt]	$\frac{1}{hour}$	eq. 44
Dewpoint temperature at 700hPa	$°C$	eq. 40
Dewpoint temperature at 850hPa	$°C$	eq. 40
K index	$°C$	eq. 45
Precipitable water	mm	eq. 46

The label, y is determined from

$$y = \begin{cases} 0, \ for\ RR < 0.1\, mmh^{-1} \\ 1, \ for\ RR \leq 0.1\, mmh^{-1} \end{cases} \tag{39}$$

$$dewpoint = 273.15 + \frac{243.5 \log(e/6.112)}{17.67 - \log(e/6.112)} \tag{40}$$

Windspeed V is defined by

$$V = \sqrt{u^2 + v^2} \tag{41}$$

where u, v is wind components in zonal and meridional direction respectively.
Wind direction is found by

$$wind\, direction = 90° - \arctan(\frac{-u}{-v}) \tag{42}$$

Moisture flux (absolute, zonal or meridional) is found by windspeed (V, u or v) multiplied with the specific humidity, q

$$moisture\ flux = V * q \tag{43}$$

The tendency variables is found by subtracting the previous time-step from the current

$$X_{tendency} = \frac{X_t - X_{t-1}}{t - (t-1)} = \frac{X_t - X_{t-1}}{hour} \tag{44}$$

where t is the current time-step, and $t-1$ is the previous time-step.

The K-index used by May et al. [2022] is defined by

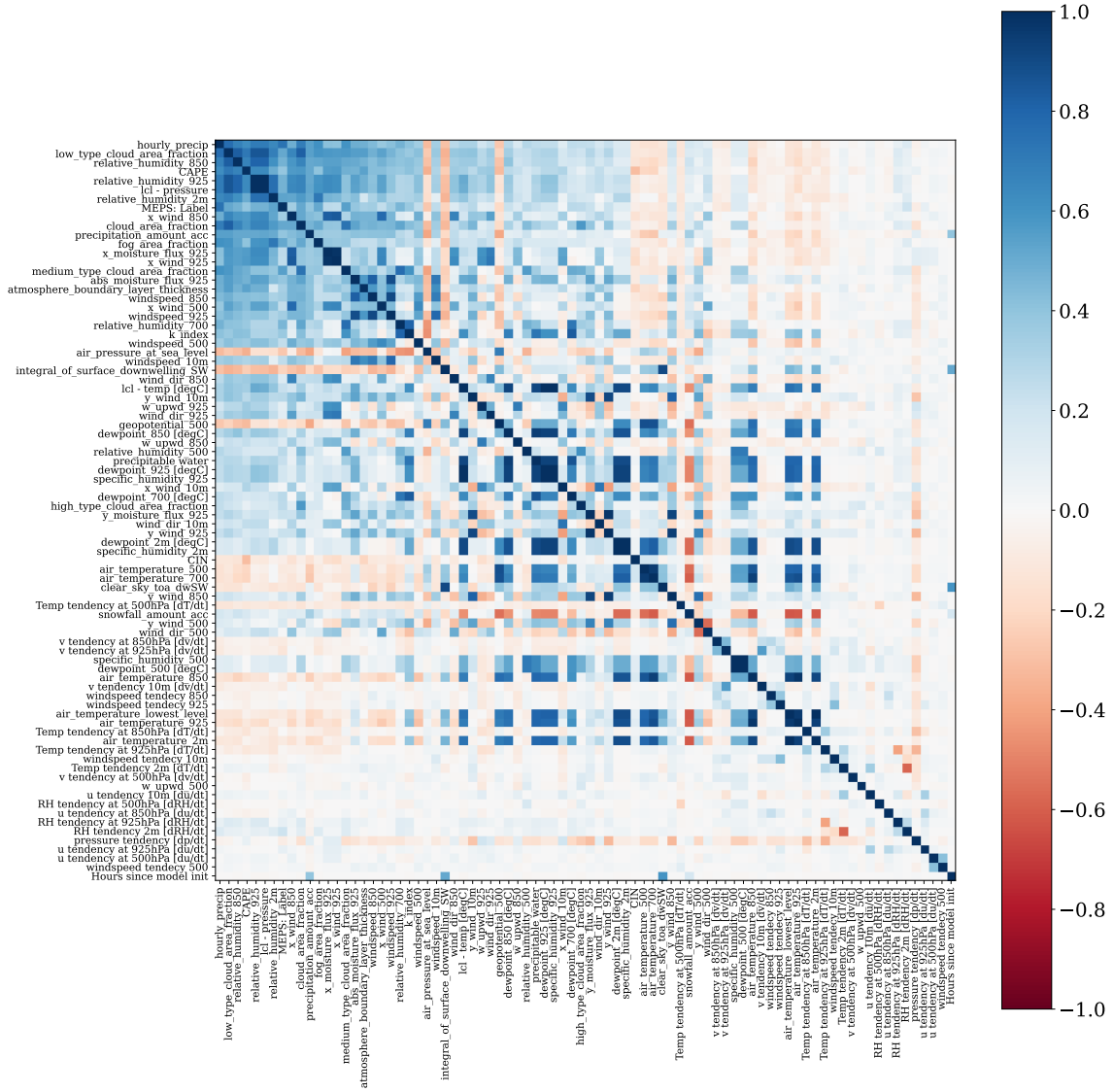
$$K = (T_{850} - T_{500}) + T_{dew,850} - (T_{700} - T_{dew,700}) \tag{45}$$

and the precipitable water is defined as

$$precipitable\, water = -\frac{1}{\rho_l g} \int_{p_{bottom}}^{p_{top}} r dp \tag{46}$$

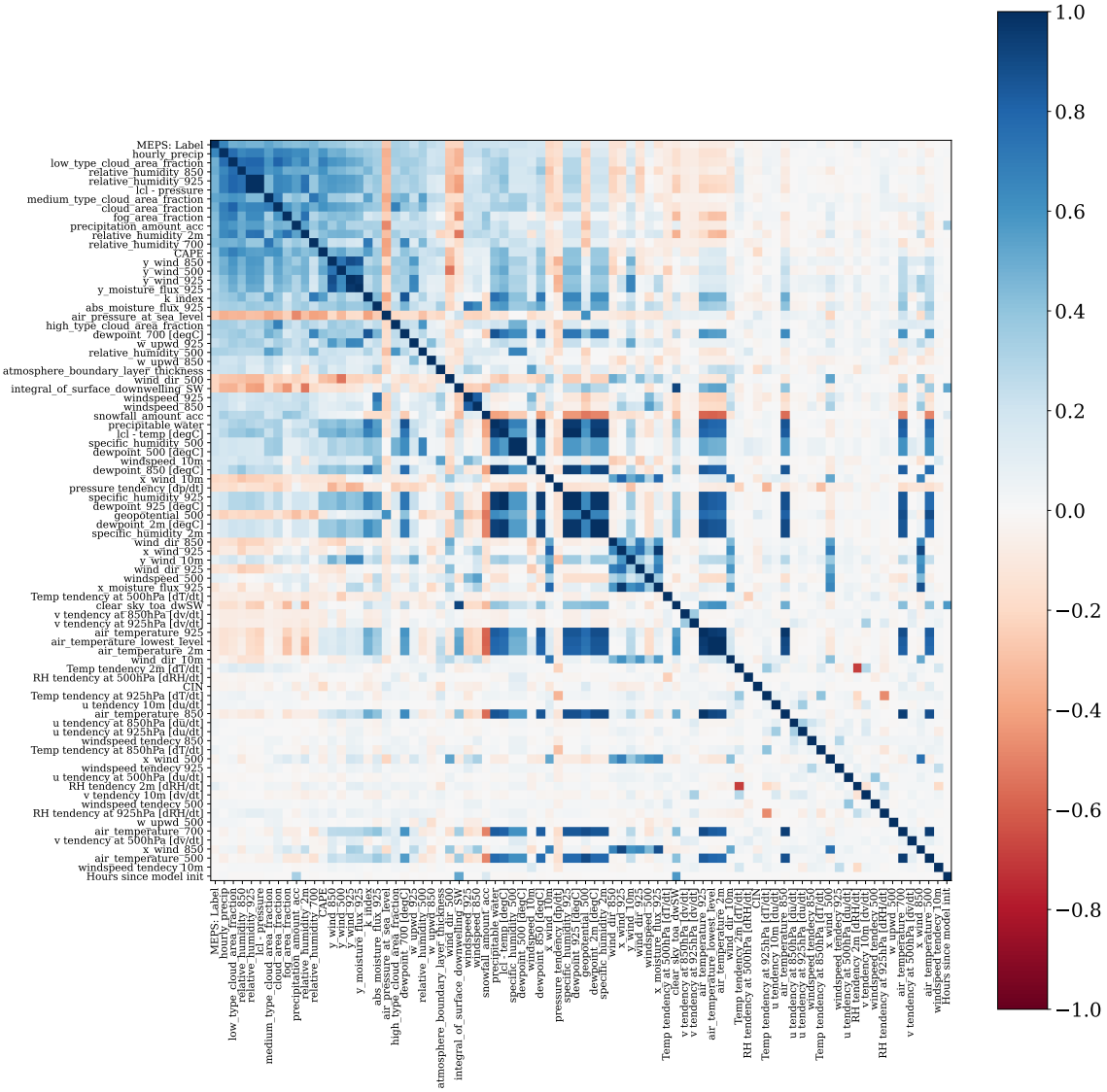
B Feature selection

B.1 Exploratory data analysis



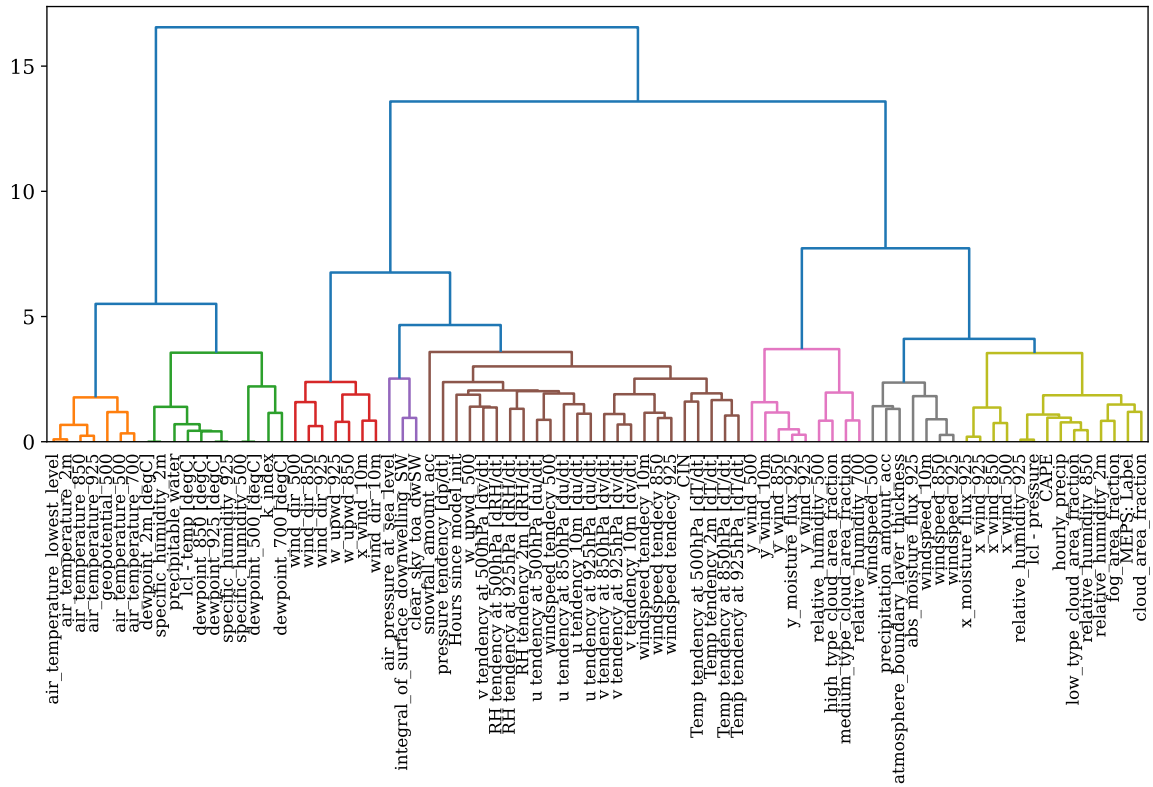
(a)

Figure B.1: Similar as fig. 16, figure shows the Spearman rank matrix for the F02 dataset.

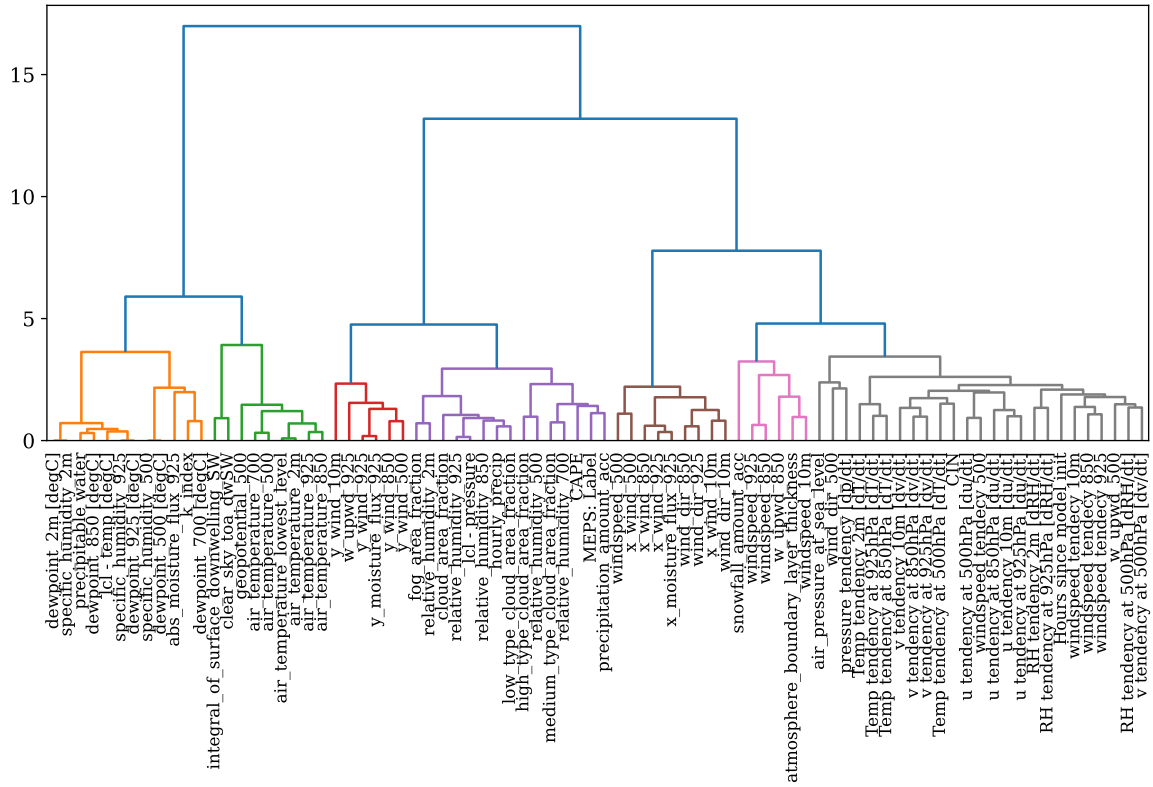


(a)

Figure B.2: Similar as fig. 16, figure shows the Spearman rank matrix for the B02 (Blindern).



(a)



(b)

Figure B.3: Similar to fig. 17, but for the 02 datasets for (a) Florida (F02) and (b) Blindern (B02).

B.2 Objective vs *n features* plot from RFECV

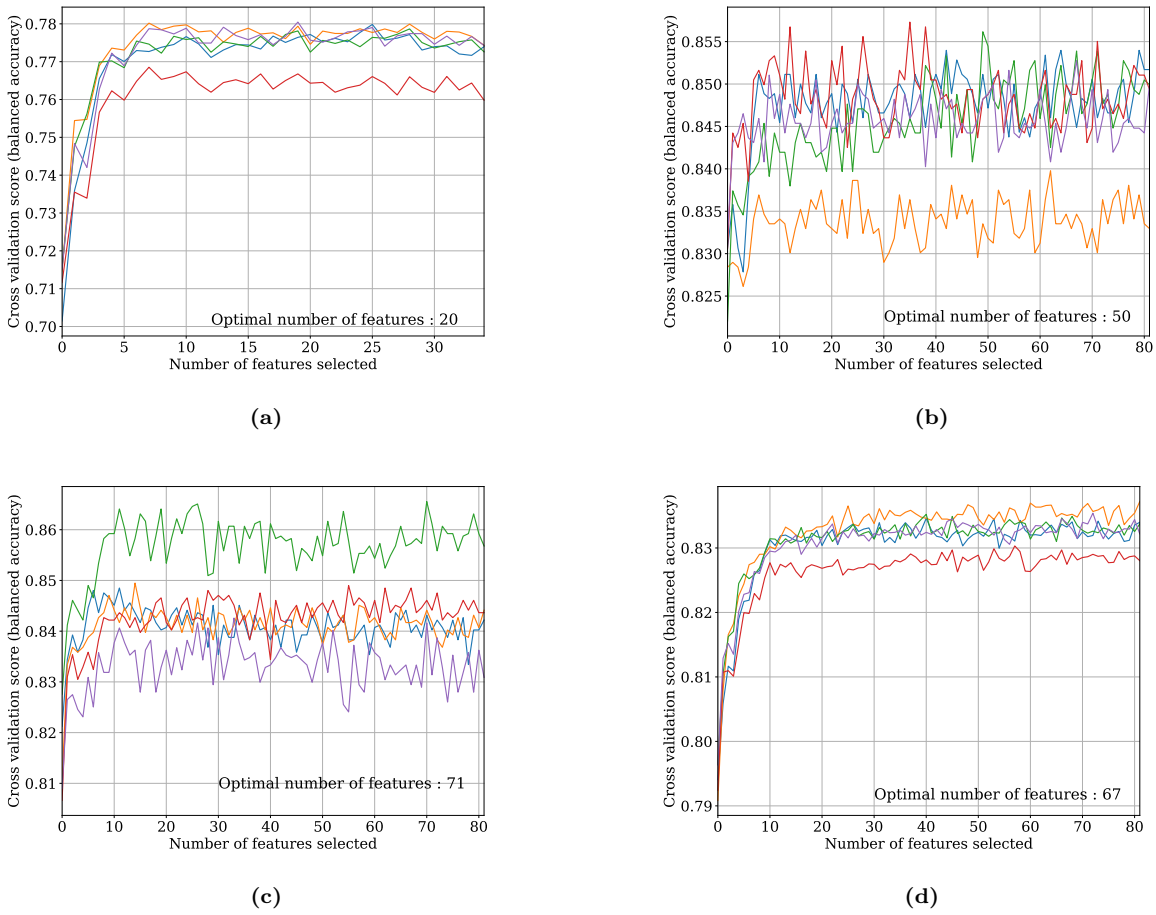


Figure B.4: Figure shows balanced accuracy plotted against number of features, n for (a) F01P02-66h. (b) F02P02-6h. (c) F02P02-12h and (d) F02P02-66h. The different coloured curves represent the 5-fold cross-validation used by the RFECV.

B.3 Feature scores and selection

Table B.3: All feature scores for F01P01 (left) and B01P01 (right). **F01P01**: The two columns to the left shows spearman rank, r_{rank}, and feature importance obtained from random forest. The two columns to the right shows features selected by scikit-learns recursive feature elimination packages, RFECV and RFE respectivly. The RFECV function uses cross validation to find the optimal number of features to include, while RFE work with a predefined number of features selected by the user (in this case $n\ features = 10$). **B01P01**: Similarly as for F01P01, the two columns to the left shows Spearman rank and feature importance. The thrid show whether the feature was selected by RFECV (marked by **x**). There was no need to use RFE for B01P01.

	F01P01				**B01P01**		
	r_{rank}	Feat imp	RFECV	RFE	r_{rank}	Feat imp	RFECV
precipitation_amount_acc	0.449	0.043	**x**	**x**	0.385	0.037	**x**
low_type_cloud_area_fraction	0.53	0.174	**x**	**x**	0.438	0.14	**x**
x_wind_850	0.468	0.098	**x**	**x**	-0.005	0.002	-
fog_area_fraction	0.438	0.042	**x**	**x**	0.389	0.043	**x**
cloud_area_fraction	0.463	0.01	**x**	**x**	0.408	0.071	**x**

Continued on next page

Table B.3: All feature scores for F01P01 (left) and B01P01 (right). **F01P01**: The two columns to the left shows spearman rank, r_{rank}, and feature importance obtained from random forest. The two columns to the right shows features selected by scikit-learns recursive feature elimination packages, RFECV and RFE respectivly. The RFECV function uses cross validation to find the optimal number of features to include, while RFE work with a predefined number of features selected by the user (in this case *n features* = 10). **B01P01**: Similarly as for F01P01, the two columns to the left shows Spearman rank and feature importance. The thrid show whether the feature was selected by RFECV (marked by **x**). There was no need to use RFE for B01P01. (Continued)

	F01P01				**B01P01**		
	r_{rank}	Feat imp	RFECV	RFE	r_{rank}	Feat imp	RFECV
medium_type_cloud_area_fraction	0.424	0.027	**x**	**x**	0.409	0.09	**x**
x_wind_925	0.424	0.074	**x**	**x**	-0.102	0.005	-
CAPE	0.495	0.101	**x**	**x**	0.353	0.03	-
relative_humidity_2m	0.49	0.086	**x**	**x**	0.362	0.062	**x**
hourly_precip	0.561	0.121	**x**	**x**	0.52	0.199	**x**
w_upwd_500	-0.031	0.0	-	-	0.008	0.0	-
w_upwd_850	0.212	0.023	**x**	-	0.218	0.022	-
w_upwd_925	0.241	0.014	**x**	-	0.242	0.012	-
relative_humidity_500	0.213	0.003	-	-	0.238	0.02	-
clear_sky_toa_dwSW	-0.111	0.001	-	-	-0.076	0.001	-
CIN	-0.126	0.01	**x**	-	-0.031	0.001	-
y_wind_850	0.107	0.002	-	-	0.352	0.057	**x**
y_wind_925	0.147	0.003	-	-	0.301	0.007	-
x_wind_500	0.352	0.004	-	-	-0.019	0.003	-
y_wind_500	0.094	0.001	-	-	0.347	0.023	-
air_temperature_500	-0.122	0.003	-	-	-0.002	0.001	-
relative_humidity_925	0.492	0.078	**x**	-	0.419	0.139	**x**
geopotential_500	-0.226	0.008	**x**	-	-0.136	0.002	-
snowfall_amount_acc	0.102	0.0	-	-	0.189	0.003	-
air_temperature_850	-0.076	0.003	-	-	-0.024	0.001	-
air_temperature_925	-0.06	0.004	**x**	-	-0.057	0.001	-
high_type_cloud_area_fraction	0.167	0.001	-	-	0.25	0.003	-
atmosphere_boundary_layer_thickness	0.38	0.025	**x**	-	0.2	0.011	-
integral_of_surface_downwelling_SW	-0.263	0.006	**x**	-	-0.192	0.001	-
y_wind_10m	0.243	0.003	-	-	0.1	0.001	-
x_wind_10m	0.193	0.007	**x**	-	-0.16	0.006	-
air_temperature_lowest_level	-0.061	0.002	-	-	-0.055	0.001	-
air_temperature_2m	-0.059	0.002	-	-	-0.051	0.001	-
air_pressure_at_sea_level	-0.282	0.019	**x**	-	-0.253	0.004	-
Hours since model init	-0.001	0.001	-	-	0.001	0.0	-

Table B.4: Same as table B.3, but for all feature scores for F02P01 (left) and B02P01 (right). It was sufficient to use RFECV for both F02P01 and B02P01, therefore RFE was used.

	F02P01			**B02P01**		
	r_{rank}	Feat imp	RFECV	r_{rank}	Feat imp	RFECV
hourly_precip	0.561	0.136	**x**	0.52	0.167	**x**
abs_moisture_flux_925	0.403	0.016	**x**	0.253	0.004	-
wind_dir_925	0.232	0.004	**x**	-0.099	0.005	**x**

Continued on next page

Table B.4: Same as table B.3, but for all feature scores for F02P01 (left) and B02P01 (right). It was sufficient to use RFECV for both F02P01 and B02P01, therefore RFE was used. (Continued)

	F02P01			B02P01		
	r_{rank}	Feat imp	RFECV	r_{rank}	Feat imp	RFECV
w_upwd_925	0.241	0.011	x	0.242	0.01	x
CIN	-0.126	0.005	x	-0.031	0.001	-
wind_dir_850	0.248	0.009	x	-0.107	0.008	x
integral_of_surface_downwelling_SW	-0.264	0.004	x	-0.192	0.001	-
air_pressure_at_sea_level	-0.281	0.007	x	-0.253	0.002	x
dewpoint_925 [degC]	0.199	0.002	x	0.147	0.001	-
k_index	0.327	0.017	x	0.255	0.008	x
low_type_cloud_area_fraction	0.53	0.109	x	0.438	0.052	x
air_temperature_850	-0.076	0.0	x	-0.024	0.0	-
specific_humidity_925	0.199	0.003	x	0.147	0.0	-
windspeed_850	0.365	0.006	x	0.189	0.001	-
w_upwd_850	0.212	0.005	x	0.218	0.013	x
x_wind_10m	0.193	0.009	x	-0.159	0.005	-
medium_type_cloud_area_fraction	0.424	0.025	x	0.41	0.085	x
x_wind_925	0.424	0.082	x	-0.102	0.001	x
x_moisture_flux_925	0.43	0.102	x	-0.079	0.001	-
fog_area_fraction	0.438	0.008	x	0.389	0.016	x
precipitation_amount_acc	0.449	0.039	x	0.385	0.034	x
cloud_area_fraction	0.463	0.016	x	0.409	0.054	x
x_wind_850	0.468	0.028	x	-0.005	0.0	-
MEPS: Label	0.488	0.089	x	0.557	0.147	x
relative_humidity_2m	0.49	0.077	x	0.362	0.053	x
lcl - pressure	0.492	0.021	x	0.418	0.052	x
relative_humidity_925	0.492	0.019	x	0.419	0.036	x
CAPE	0.495	0.063	x	0.353	0.003	x
relative_humidity_850	0.503	0.044	x	0.431	0.085	x
geopotential_500	-0.226	0.004	x	-0.136	0.001	-
air_temperature_2m	-0.06	0.002	-	-0.051	0.001	-
Temp tendency at 850hPa [dT/dt]	-0.061	0.0	-	0.019	0.0	-
v tendency 10m [dv/dt]	-0.07	0.0	-	-0.015	0.0	-
air_temperature_925	-0.061	0.001	-	-0.057	0.0	-
air_temperature_lowest_level	-0.062	0.0	-	-0.055	0.0	-
windspeed tendecy 925	-0.063	0.0	-	-0.018	0.0	-
windspeed tendecy 850	-0.07	0.0	-	-0.021	0.0	-
dewpoint_500 [degC]	0.08	0.0	-	0.164	0.001	-
specific_humidity_500	0.08	0.0	-	0.164	0.001	-
Temp tendency at 925hPa [dT/dt]	-0.042	0.0	-	0.03	0.0	-
RH tendency at 500hPa [dRH/dt]	-0.027	0.0	-	-0.033	0.0	-
windspeed tendecy 10m	-0.04	0.0	-	-0.002	0.0	-
Temp tendency 2m [dT/dt]	0.04	0.0	-	0.042	0.0	-
v tendency at 500hPa [dv/dt]	-0.039	0.0	-	0.005	0.0	-
w_upwd_500	-0.03	0.0	-	0.008	0.0	-
u tendency 10m [du/dt]	0.029	0.0	-	0.027	0.0	-
v tendency at 850hPa [dv/dt]	-0.08	0.0	-	-0.072	0.001	-
u tendency at 850hPa [du/dt]	-0.025	0.0	-	0.023	0.0	-

Continued on next page

Table B.4: Same as table B.3, but for all feature scores for F02P01 (left) and B02P01 (right). It was sufficient to use RFECV for both F02P01 and B02P01, therefore RFE was used. (Continued)

	F02P01			**B02P01**		
	r_{rank}	Feat imp	RFECV	r_{rank}	Feat imp	RFECV
RH tendency at 925hPa [dRH/dt]	0.023	0.0	-	0.009	0.0	-
RH tendency 2m [dRH/dt]	0.021	0.001	-	0.016	0.002	**x**
pressure tendency [dp/dt]	0.006	0.0	-	-0.151	0.001	-
u tendency at 925hPa [du/dt]	-0.004	0.001	-	0.023	0.0	-
u tendency at 500hPa [du/dt]	-0.004	0.0	-	-0.016	0.0	-
v tendency at 925hPa [dv/dt]	-0.08	0.0	-	-0.061	0.0	-
high_type_cloud_area_fraction	0.166	0.0	-	0.25	0.002	-
wind_dir_500	0.083	0.0	-	-0.196	0.006	**x**
y_wind_500	0.094	0.0	-	0.347	0.013	**x**
atmosphere_boundary_layer_thickness	0.38	0.002	-	0.2	0.003	**x**
x_wind_500	0.351	0.001	-	-0.019	0.001	-
windspeed_925	0.35	0.002	-	0.191	0.0	-
relative_humidity_700	0.334	0.004	-	0.361	0.055	**x**
windspeed_500	0.315	0.001	-	0.088	0.001	-
windspeed_10m	0.277	0.003	-	0.161	0.0	-
lcl - temp [degC]	0.247	0.002	-	0.187	0.002	-
y_wind_10m	0.243	0.001	-	0.101	0.0	-
dewpoint_850 [degC]	0.217	0.002	-	0.16	0.0	-
relative_humidity_500	0.212	0.0	-	0.238	0.005	**x**
precipitable water	0.201	0.003	-	0.188	0.001	-
dewpoint_700 [degC]	0.182	0.002	-	0.248	0.003	**x**
y_moisture_flux_925	0.161	0.001	-	0.288	0.006	**x**
wind_dir_10m	0.155	0.003	-	-0.048	0.001	-
y_wind_925	0.147	0.0	-	0.301	0.009	**x**
dewpoint_2m [degC]	0.132	0.001	-	0.109	0.0	-
specific_humidity_2m	0.132	0.001	-	0.109	0.0	-
air_temperature_500	-0.123	0.003	-	-0.002	0.001	-
air_temperature_700	-0.111	0.001	-	0.006	0.002	-
clear_sky_toa_dwSW	-0.111	0.001	-	-0.076	0.0	-
y_wind_850	0.107	0.001	-	0.352	0.028	**x**
Temp tendency at 500hPa [dT/dt]	-0.107	0.0	-	-0.078	0.0	-
snowfall_amount_acc	0.102	0.0	-	0.189	0.002	-
windspeed tendecy 500	-0.002	0.0	-	0.013	0.0	-

Table B.5: Same as table B.3, but for all feature scores for F01P02 (i.e F01P02-6h, F01P02-12h, F01P02-66h) and B01P02 (i.e B01P02-6h, B01P02-12h, B01P02-66h). For F01P02-66h it was neccisary to use RFE, where *n features* = 30).

	F01P02-6h			**F01P02-12h**			**F01P02-6h**				**B01P02-6h**			**B01P02-12h**			**B01P02-66h**		
	Spearman	Feat imp	RFECV	Spearman	Feat imp	RFECV	Spearman	Feat imp	RFECV	RFE	Spearman	Feat imp	RFECV	Spearman	Feat imp	RFECV	Spearman	Feat imp	RFECV
precipitation_amount_acc	0.64	0.24	x	0.62	0.23	x	0.31	0.05	x	x	0.56	0.22	x	0.52	0.2	x	0.26	0.04	x
low_type_cloud_area_fraction	0.57	0.12	x	0.56	0.12	x	0.31	0.16	x	x	0.46	0.12	-	0.45	0.13	-	0.25	0.13	x
hourly_precip	0.63	0.08	x	0.6	0.09	x	0.46	0.11	x	x	0.57	0.15	-	0.56	0.16	x	0.36	0.18	x
x_wind_850	0.48	0.08	x	0.48	0.1	x	0.36	0.1	x	x	-0.0	0.0	-	-0.0	0.0	-	0.0	0.0	-
relative_humidity_2m	0.54	0.14	x	0.52	0.12	x	0.31	0.1	x	x	0.38	0.05	-	0.38	0.05	-	0.19	0.06	x
CAPE	0.5	0.05	x	0.51	0.07	x	0.2	0.1	x	x	0.36	0.02	-	0.37	0.03	-	0.19	0.03	-
fog_area_fraction	0.49	0.04	x	0.48	0.04	x	0.23	0.03	x	x	0.42	0.03	-	0.42	0.03	-	0.22	0.04	x
cloud_area_fraction	0.5	0.04	x	0.48	0.01	x	0.23	0.02	x	x	0.43	0.06	-	0.43	0.06	-	0.22	0.06	x
x_wind_500	0.35	0.0	-	0.35	0.0	-	0.26	0.0	-	-	-0.02	0.0	-	-0.02	0.0	-	-0.03	0.0	-
y_wind_925	0.14	0.0	-	0.13	0.0	-	0.17	0.0	-	-	0.31	0.0	-	0.31	0.0	-	0.18	0.0	-
y_wind_850	0.1	0.0	-	0.1	0.0	-	0.13	0.0	-	-	0.36	0.06	-	0.36	0.05	-	0.18	0.06	x
w_upwd_925	0.27	0.01	-	0.27	0.02	-	0.2	0.02	x	-	0.25	0.01	-	0.24	0.02	-	0.13	0.01	-
y_wind_500	0.09	0.0	-	0.09	0.0	-	0.14	0.0	-	-	0.35	0.02	-	0.35	0.02	-	0.18	0.02	-
relative_humidity_500	0.22	0.0	-	0.21	0.0	-	0.18	0.0	-	-	0.26	0.02	-	-0.26	0.02	-	-0.16	0.02	-
w_upwd_850	0.24	0.02	-	0.24	0.02	-	0.22	0.02	x	-	0.22	0.02	-	0.23	0.02	-	-0.13	0.02	-
w_upwd_500	-0.04	0.0	-	-0.04	0.0	-	0.03	0.0	-	-	0.01	0.0	-	-0.02	0.0	-	0.03	0.0	-
clear_sky_toa_dwSW	-0.11	0.0	-	-0.13	0.0	-	-0.1	0.0	-	-	-0.04	0.0	-	-0.06	0.0	-	0.03	0.0	-
CIN	-0.12	0.0	-	-0.12	0.01	-	-0.14	0.01	x	-	-0.01	0.0	-	0.01	0.0	-	-0.02	0.0	-
x_wind_925	0.44	0.05	-	0.44	0.05	x	0.31	0.07	x	x	-0.1	0.0	-	-0.1	0.0	-	0.06	0.0	-
air_temperature_500	-0.12	0.0	-	-0.12	0.0	-	-0.02	0.0	-	-	0.0	0.0	-	-0.0	0.0	-	0.01	0.0	-
relative_humidity_925	0.53	0.06	-	0.52	0.07	x	0.29	0.09	x	-	0.44	0.09	-	0.43	0.08	-	0.23	0.16	x
geopotential_500	-0.22	0.0	-	-0.23	0.0	-	-0.13	0.01	x	-	-0.14	0.0	-	-0.14	0.0	-	-0.06	0.0	-
snowfall_amount_acc	0.14	0.0	-	0.13	0.0	-	-0.01	0.0	-	-	0.26	0.01	-	0.25	0.01	-	0.15	0.01	-
air_temperature_850	-0.07	0.0	-	-0.08	0.0	-	0.01	0.0	-	-	-0.02	0.0	-	-0.03	0.0	-	-0.03	0.0	-
air_temperature_925	-0.06	0.0	-	-0.06	0.0	-	0.0	0.0	x	-	-0.06	0.0	-	-0.06	0.0	-	0.05	0.0	-
medium_type_cloud_area_fraction	0.44	0.01	-	0.43	0.01	x	0.31	0.03	x	x	0.44	0.1	-	0.44	0.1	-	0.24	0.1	x
high_type_cloud_area_fraction	0.16	0.0	-	0.17	0.0	-	0.16	0.0	-	-	0.25	0.0	-	0.25	0.0	-	0.15	0.0	-
atmosphere_boundary_layer_thickness	0.38	0.02	-	0.39	0.02	-	0.28	0.03	x	-	0.21	0.01	-	-0.23	0.01	-	0.09	0.02	-
integral_of_surface_downwelling_SW	-0.24	0.0	-	-0.32	0.01	-	-0.17	0.01	x	-	-0.15	0.0	-	-0.16	0.0	-	-0.07	0.0	-
y_wind_10m	0.25	0.0	-	0.24	0.0	-	0.22	0.0	-	-	0.1	0.0	-	0.1	0.0	-	0.06	0.0	-
x_wind_10m	0.2	0.0	-	0.21	0.0	-	0.17	0.01	x	-	-0.17	0.0	-	0.21	0.0	-	-0.07	0.0	-
air_temperature_lowest_level	-0.07	0.0	-	-0.07	0.0	-	-0.01	0.0	-	-	-0.06	0.0	-	-0.06	0.0	-	0.06	0.0	-
air_temperature_2m	-0.07	0.0	-	-0.06	0.0	-	-0.01	0.0	-	-	-0.06	0.0	-	-0.06	0.0	-	0.04	0.0	-
air_pressure_at_sea_level	-0.28	0.01	-	-0.28	0.01	-	-0.22	0.02	x	-	-0.26	0.0	-	0.26	0.0	-	0.15	0.01	-

Continued on next page

Table B.5: Same as table B.3, but for all feature scores for F01P02 (i.e F01P02-6h, F01P02-12h, F01P02-66h) and B01P02 (i.e B01P02-6h, B01P02-12h, B01P02-66h). For F01P02-66h it was neccisary to use RFE, where *n features* = 30). (Continued)

	F01P02-6h			**F01P02-12h**			**F01P02-6h**				**B01P02-6h**			**B01P02-12h**			**B01P02-66h**		
	Spearman	Feat imp	RFECV	Spearman	Feat imp	RFECV	Spearman	Feat imp	RFECV	RFE	Spearman	Feat imp	RFECV	Spearman	Feat imp	RFECV	Spearman	Feat imp	RFECV
Hours since model init	0.0	0.0	-	0.0	0.0	-	-0.0	0.0	-	-	-0.0	0.0	-	0.0	0.0	-	0.0	0.0	-

Table B.6: Same as table B.3, but for all feature scores for F02P02 (i.e F02P02-6h, F02P02-12h, F02P02-66h) and B02P02 (i.e B02P02-6h, B02P02-12h, B02P02-66h). It was neccisary to do the extra step of RFE for all subsets of F02P02, with $n\ features = 30$).

	F02P02-6h				F02P02-12h				F02P02-66h				B02P02-6h			B02P02-12h			B02P02-66h		
	Spearman	Feat imp	RFECV	RFE	Spearman	Feat imp	RFECV	RFE	Spearman	Feat imp	RFECV	RFE	Spearman	Feat imp	RFECV	Spearman	Feat imp	RFECV	Spearman	Feat imp	RFECV
precipitation_amount_acc	0.64	0.14	x	x	0.62	0.14	x	x	0.46	0.04	x	x	0.56	0.19	x	0.52	0.15	x	0.25	0.02	x
k_index	0.34	0.02	x	x	0.33	0.02	x	x	0.23	0.02	x	x	0.27	0.0	-	0.26	0.0	-	0.18	0.0	x
relative_humidity_850	0.54	0.05	x	x	0.52	0.04	x	x	0.31	0.05	x	x	0.45	0.04	-	0.45	0.08	x	0.21	0.07	x
MEPS: Label	0.55	0.08	x	x	0.53	0.07	x	x	0.36	0.1	x	x	0.65	0.12	-	0.63	0.15	x	0.34	0.16	x
hourly_precip	0.63	0.13	x	x	0.6	0.14	x	x	0.39	0.13	x	x	0.57	0.15	-	0.56	0.14	x	0.36	0.18	x
CAPE	0.5	0.04	x	x	0.51	0.06	x	x	0.31	0.06	x	x	0.36	0.0	-	0.37	0.01	-	0.15	0.0	x
windspeed_10m	0.27	0.0	x	x	0.28	0.0	x	-	-0.22	0.0	x	-	0.18	0.0	-	0.17	0.0	-	0.07	0.0	-
w_upwd_925	0.27	0.01	x	x	-0.28	0.01	x	x	0.22	0.01	x	x	0.25	0.01	-	0.26	0.02	-	0.18	0.01	x
windspeed_925	0.35	0.0	x	x	0.34	0.0	x	-	0.23	0.0	x	-	0.19	0.0	-	0.2	0.0	-	0.12	0.0	-
windspeed_850	0.37	0.0	x	x	0.37	0.0	x	x	0.25	0.0	x	x	0.19	0.0	-	0.2	0.0	-	0.13	0.0	-
dewpoint_700 [degC]	0.19	0.0	x	x	0.18	0.0	x	-	0.17	0.0	x	-	0.26	0.0	-	0.26	0.0	-	0.18	0.0	x
x_wind_850	0.48	0.02	x	x	0.48	0.03	x	x	0.28	0.03	x	x	-0.0	0.0	-	0.0	0.0	-	-0.02	0.0	-
x_wind_925	0.44	0.08	x	x	0.43	0.06	x	x	0.27	0.08	x	x	-0.1	0.0	-	-0.1	0.0	-	-0.07	0.0	-
relative_humidity_500	0.22	0.0	x	x	0.21	0.0	x	x	0.18	0.0	x	-	0.26	0.0	-	0.24	0.0	-	0.13	0.01	x
dewpoint_925 [degC]	0.21	0.0	x	x	0.21	0.0	x	-	-0.17	0.0	x	-	0.15	0.0	-	0.15	0.0	-	0.15	0.0	-
geopotential_500	-0.23	0.0	x	x	-0.23	0.0	x	x	0.18	0.01	x	x	-0.14	0.0	-	-0.14	0.0	-	-0.03	0.0	-
relative_humidity_925	0.53	0.02	x	x	0.52	0.02	x	x	0.31	0.02	x	x	0.44	0.04	-	0.44	0.04	-	0.22	0.04	x
relative_humidity_700	0.36	0.0	x	x	0.35	0.0	x	x	0.25	0.0	x	x	0.4	0.05	-	0.38	0.04	x	0.19	0.05	x
relative_humidity_2m	0.54	0.08	x	x	0.53	0.1	x	x	0.35	0.07	x	x	0.39	0.03	-	0.38	0.04	-	0.19	0.04	x
fog_area_fraction	0.49	0.01	x	x	0.48	0.01	x	x	0.29	0.01	x	x	0.43	0.01	-	0.42	0.01	-	0.05	0.02	x
cloud_area_fraction	0.5	0.05	x	x	0.48	0.02	x	x	0.31	0.02	x	x	0.43	0.06	-	0.43	0.06	x	0.18	0.06	x
medium_type_cloud_area_fraction	0.44	0.02	x	x	0.44	0.02	x	x	0.28	0.02	x	x	0.44	0.08	-	0.43	0.09	x	0.26	0.08	x
low_type_cloud_area_fraction	0.57	0.1	x	x	0.56	0.1	x	x	0.36	0.1	x	x	0.46	0.08	-	0.45	0.06	x	0.24	0.07	x
lcl - pressure	0.53	0.03	x	x	0.52	0.03	x	x	0.31	0.02	x	x	0.44	0.04	-	0.43	0.02	x	0.21	0.05	x
air_pressure_at_sea_level	-0.28	0.0	x	x	-0.32	0.0	x	x	0.22	0.01	x	x	-0.26	0.0	-	-0.26	0.0	-	-0.16	0.0	x
atmosphere_boundary_layer_thickness	0.38	0.01	x	x	0.39	0.01	x	-	0.26	0.0	x	x	0.21	0.0	-	0.21	0.0	-	0.15	0.0	-
wind_dir_925	0.25	0.0	x	x	0.25	0.0	x	x	0.2	0.0	x	x	-0.11	0.0	-	-0.11	0.0	-	-0.03	0.0	x
precipitable water	0.21	0.0	x	x	0.21	0.0	x	x	0.17	0.0	x	x	0.19	0.0	-	0.19	0.0	-	0.19	0.0	-
x_wind_10m	0.2	0.0	x	x	0.2	0.0	x	x	0.17	0.01	x	x	-0.17	0.0	-	-0.16	0.0	-	-0.07	0.0	-
x_moisture_flux_925	0.45	0.06	x	x	0.45	0.06	x	x	0.28	0.09	x	x	-0.08	0.0	-	-0.08	0.0	-	-0.07	0.0	-
windspeed tendecy 500	-0.01	0.0	-	-	-0.0	0.0	-	-	0.0	0.0	-	-	0.02	0.0	-	0.01	0.0	-	0.01	0.0	-
u tendency at 850hPa [du/dt]	-0.01	0.0	-	-	0.01	0.0	-	-	-0.0	0.0	x	-	0.03	0.0	-	0.02	0.0	-	0.04	0.0	-
RH tendency 2m [dRH/dt]	0.02	0.0	x	-	0.02	0.0	-	-	0.0	0.0	x	-	0.03	0.0	-	0.02	0.0	-	0.01	0.0	-
u tendency at 925hPa [du/dt]	0.01	0.0	-	-	-0.01	0.0	x	x	0.0	0.0	x	-	0.05	0.0	-	0.04	0.0	-	0.04	0.0	-

Continued on next page

Table B.6: Same as table B.3, but for all feature scores for F02P02 (i.e F02P02-6h, F02P02-12h, F02P02-66h) and B02P02 (i.e B02P02-6h, B02P02-12h, B02P02-66h). It was neccisary to do the extra step of RFE for all subsets of F02P02, with *n features* = 30). (Continued)

	F02P02-6h				**F02P02-12h**				**F02P02-66h**				**B02P02-6h**			**B02P02-12h**			**B02P02-66h**		
	Spearman	Feat imp	RFECV	RFE	Spearman	Feat imp	RFECV	RFE	Spearman	Feat imp	RFECV	RFE	Spearman	Feat imp	RFECV	Spearman	Feat imp	RFECV	Spearman	Feat imp	RFECV
pressure tendency [dp/dt]	0.03	0.0	x	-	0.03	0.0	-	-	-0.01	0.0	x	-	-0.14	0.0	-	-0.15	0.0	-	-0.11	0.0	-
v tendency 10m [dv/dt]	-0.08	0.0	-	-	0.08	0.0	x	-	0.06	0.0	x	-	0.0	0.0	-	-0.01	0.0	-	0.0	0.0	-
u tendency 10m [du/dt]	0.04	0.0	x	-	-0.04	0.0	-	-	0.01	0.0	-	-	0.04	0.0	-	0.04	0.0	-	0.03	0.0	-
Temp tendency 2m [dT/dt]	0.05	0.0	-	-	-0.06	0.0	x	-	-0.02	0.0	x	-	0.04	0.0	-	0.04	0.0	-	-0.0	0.0	-
Temp tendency at 500hPa [dT/dt]	-0.13	0.0	-	-	-0.13	0.0	x	-	-0.13	0.0	-	-	-0.09	0.0	-	-0.08	0.0	-	-0.05	0.0	-
u tendency at 500hPa [du/dt]	-0.0	0.0	-	-	0.0	0.0	x	-	-0.0	0.0	-	-	-0.0	0.0	-	-0.02	0.0	-	0.0	0.0	-
v tendency at 925hPa [dv/dt]	-0.1	0.0	-	-	0.1	0.0	x	-	0.11	0.0	-	-	-0.06	0.0	-	-0.06	0.0	-	-0.04	0.0	-
v tendency at 850hPa [dv/dt]	-0.1	0.0	-	-	-0.1	0.0	x	-	0.11	0.0	-	-	-0.08	0.0	-	-0.08	0.0	-	-0.05	0.0	-
v tendency at 500hPa [dv/dt]	-0.04	0.0	-	-	0.04	0.0	-	-	-0.01	0.0	-	-	0.0	0.0	-	-0.0	0.0	-	0.01	0.0	-
Temp tendency at 925hPa [dT/dt]	-0.03	0.0	-	-	0.03	0.0	x	-	0.01	0.0	-	-	0.03	0.0	-	0.02	0.0	-	0.0	0.0	-
Temp tendency at 850hPa [dT/dt]	-0.07	0.0	x	-	-0.07	0.0	x	-	0.05	0.0	-	-	0.02	0.0	-	0.02	0.0	-	0.0	0.0	-
windspeed tendecy 925	-0.08	0.0	x	-	-0.08	0.0	x	-	0.07	0.0	-	-	-0.0	0.0	-	-0.01	0.0	-	-0.02	0.0	-
RH tendency at 925hPa [dRH/dt]	0.03	0.0	-	-	-0.03	0.0	x	-	-0.01	0.0	x	-	0.02	0.0	-	0.0	0.0	-	0.0	0.0	-
RH tendency at 500hPa [dRH/dt]	-0.04	0.0	x	-	-0.04	0.0	x	-	-0.01	0.0	-	-	-0.05	0.0	-	-0.04	0.0	-	-0.03	0.0	-
dewpoint_850 [degC]	0.23	0.0	-	-	0.23	0.0	x	x	0.18	0.0	x	-	0.17	0.0	-	0.17	0.0	-	0.15	0.0	-
lcl - temp [degC]	0.26	0.0	x	-	0.27	0.0	x	-	0.21	0.0	x	x	0.19	0.0	-	0.19	0.0	-	0.17	0.0	-
abs_moisture_flux_925	0.41	0.02	x	-	0.41	0.02	x	x	0.26	0.02	x	x	0.26	0.0	-	0.26	0.0	-	0.24	0.0	-
y_moisture_flux_925	0.16	0.0	-	-	0.16	0.0	-	-	-0.16	0.0	x	-	0.29	0.0	-	0.29	0.0	-	0.23	0.0	-
windspeed tendecy 850	-0.08	0.0	-	-	0.09	0.0	x	-	-0.07	0.0	-	-	-0.03	0.0	-	-0.03	0.0	-	-0.02	0.0	-
dewpoint_2m [degC]	0.15	0.0	x	-	0.14	0.0	x	-	0.14	0.0	x	-	0.11	0.0	-	0.11	0.0	-	0.13	0.0	-
windspeed tendecy 10m	-0.04	0.0	-	-	-0.05	0.0	x	-	-0.01	0.0	-	-	-0.0	0.0	-	-0.01	0.0	-	0.0	0.0	-
wind_dir_500	0.08	0.0	x	-	-0.09	0.0	x	-	-0.08	0.0	x	-	-0.2	0.0	-	-0.19	0.0	-	-0.07	0.0	x
air_temperature_2m	-0.07	0.0	x	-	-0.07	0.0	x	-	-0.03	0.0	x	-	-0.06	0.0	-	-0.06	0.0	-	0.03	0.0	-
air_temperature_lowest_level	-0.07	0.0	x	-	-0.07	0.0	x	-	-0.04	0.0	x	-	-0.06	0.0	-	-0.06	0.0	-	0.03	0.0	-
y_wind_10m	0.25	0.0	x	-	0.26	0.0	x	-	0.2	0.0	x	-	0.1	0.0	-	0.1	0.0	-	0.09	0.0	-
integral_of_surface_downwelling_SW	-0.24	0.0	x	-	0.24	0.0	x	x	0.18	0.01	x	x	-0.15	0.0	-	-0.23	0.0	-	-0.06	0.0	-
high_type_cloud_area_fraction	0.16	0.0	-	-	0.17	0.0	x	-	0.16	0.0	x	-	0.25	0.0	-	0.26	0.0	-	0.15	0.0	-
air_temperature_925	-0.06	0.0	x	-	-0.06	0.0	x	-	0.03	0.0	x	-	-0.06	0.0	-	-0.06	0.0	-	0.04	0.0	-
air_temperature_850	-0.07	0.0	-	-	-0.07	0.0	x	-	-0.06	0.0	x	-	-0.02	0.0	-	-0.02	0.0	-	0.06	0.0	-
air_temperature_500	-0.12	0.0	x	-	-0.12	0.0	x	-	0.12	0.0	x	-	0.0	0.0	-	-0.0	0.0	-	0.06	0.0	-
x_wind_500	0.35	0.0	-	-	0.35	0.0	x	-	0.24	0.0	x	-	-0.02	0.0	-	-0.02	0.0	-	-0.03	0.0	-
y_wind_925	0.14	0.0	-	-	0.13	0.0	x	-	-0.13	0.0	x	-	0.31	0.0	-	0.31	0.01	-	0.18	0.01	x
y_wind_850	0.1	0.0	-	-	-0.09	0.0	x	-	-0.1	0.0	x	-	0.36	0.02	-	0.36	0.02	-	0.23	0.03	x
y_wind_500	0.09	0.0	x	-	0.09	0.0	x	-	-0.09	0.0	x	-	0.35	0.01	-	0.35	0.01	-	0.22	0.01	x

Continued on next page

Table B.6: Same as table B.3, but for all feature scores for F02P02 (i.e F02P02-6h, F02P02-12h, F02P02-66h) and B02P02 (i.e B02P02-6h, B02P02-12h, B02P02-66h). It was neccisary to do the extra step of RFE for all subsets of F02P02, with $n\ features = 30$). (Continued)

	F02P02-6h				**F02P02-12h**				**F02P02-66h**				**B02P02-6h**			**B02P02-12h**			**B02P02-66h**		
	Spearman	Feat imp	RFECV	RFE	Spearman	Feat imp	RFECV	RFE	Spearman	Feat imp	RFECV	RFE	Spearman	Feat imp	RFECV	Spearman	Feat imp	RFECV	Spearman	Feat imp	RFECV
w_upwd_850	0.24	0.01	-	-	0.24	0.01	x	-	0.19	0.0	x	x	0.22	0.01	-	0.23	0.0	-	0.19	0.0	x
w_upwd_500	-0.04	0.0	-	-	-0.04	0.0	x	-	0.01	0.0	-	-	0.01	0.0	-	0.01	0.0	-	0.0	0.0	-
clear_sky_toa_dwSW	-0.11	0.0	-	-	-0.11	0.0	-	-	0.12	0.0	x	-	-0.04	0.0	-	-0.06	0.0	-	0.01	0.0	-
CIN	-0.12	0.0	-	-	-0.12	0.0	x	-	0.12	0.01	x	x	-0.01	0.0	-	-0.03	0.0	-	-0.13	0.0	-
Hours since model init	0.0	0.0	-	-	-0.0	0.0	-	-	-0.0	0.0	-	-	-0.0	0.0	-	-0.0	0.0	-	0.0	0.0	-
air_temperature_700	-0.11	0.0	-	-	-0.12	0.0	-	-	-0.12	0.0	x	-	0.01	0.0	-	0.01	0.0	-	0.07	0.0	-
dewpoint_500 [degC]	0.08	0.0	-	-	-0.08	0.0	-	-	0.06	0.0	x	-	0.17	0.0	-	0.16	0.0	-	0.13	0.0	-
snowfall_amount_acc	0.14	0.0	-	-	0.13	0.0	x	-	0.13	0.0	x	-	0.26	0.03	-	0.25	0.02	-	0.06	0.0	-
specific_humidity_925	0.21	0.0	x	-	0.21	0.0	x	-	-0.17	0.0	x	-	0.15	0.0	-	0.15	0.0	-	0.16	0.0	-
specific_humidity_500	0.08	0.0	-	-	0.08	0.0	x	-	-0.06	0.0	x	-	0.17	0.0	-	0.16	0.0	-	0.15	0.0	-
windspeed_500	0.32	0.0	x	-	0.32	0.0	x	-	0.23	0.0	x	-	0.09	0.0	-	0.08	0.0	-	0.06	0.0	-
wind_dir_10m	0.16	0.0	-	-	0.15	0.0	x	-	0.14	0.0	x	-	-0.06	0.0	-	-0.05	0.0	-	-0.02	0.0	-
wind_dir_850	0.25	0.01	x	-	0.26	0.01	x	x	0.2	0.01	x	-	-0.11	0.0	-	-0.11	0.0	-	-0.03	0.01	x
specific_humidity_2m	0.15	0.0	-	-	0.14	0.0	x	-	-0.14	0.0	x	-	0.11	0.0	-	0.11	0.0	-	0.14	0.0	-

B.4 Combined dataset

Table B.7: Features scores for the combined dataset FB. Spearman rank, r_{rank}, and feature importance is the two left columns, while which features were selected by RFECV and RFE is shown in the two columns to the right. Note that both zonal and meridional wind was selected by the RFECV, but only zonal for the RFE. This emphasises the need to feature engineer location specific features (such as wind which is very influenced by local topography) in order to train on a combined dataset.

	[FB] Spearman rank	[FB] Feature Importance	[FB] RFECV	[FB] RFE
precipitation_amount_acc	0.375	0.054	**x**	**x**
atmosphere_boundary_layer_thickness	0.272	0.032	**x**	**x**
x_wind_925	0.207	0.022	**x**	**x**
relative_humidity_925	0.384	0.092	**x**	**x**
air_temperature_925	-0.046	0.008	**x**	**x**
cloud_area_fraction	0.374	0.037	**x**	**x**
medium_type_cloud_area_fraction	0.351	0.035	**x**	**x**
low_type_cloud_area_fraction	0.418	0.17	**x**	**x**
CAPE	0.393	0.088	**x**	**x**
x_wind_10m	0.073	0.007	**x**	**x**
CIN	-0.072	0.01	**x**	**x**
relative_humidity_2m	0.383	0.129	**x**	**x**
air_temperature_2m	-0.04	0.009	**x**	**x**
hourly_precip	0.467	0.147	**x**	**x**
x_wind_850	0.261	0.033	**x**	**x**
clear_sky_toa_dwSW	-0.084	0.0	-	-
w_upwd_500	-0.004	0.0	-	-
x_wind_500	0.2	0.004	-	-
w_upwd_850	0.157	0.018	**x**	-
w_upwd_925	0.179	0.013	**x**	-
y_wind_500	0.144	0.002	-	-
y_wind_850	0.162	0.005	-	-
y_wind_925	0.172	0.006	**x**	-
air_temperature_500	-0.067	0.003	-	-
relative_humidity_500	0.177	0.002	-	-
geopotential_500	-0.164	0.006	**x**	-
snowfall_amount_acc	0.108	0.001	-	-
air_temperature_850	-0.046	0.004	**x**	-
fog_area_fraction	0.347	0.037	**x**	-
high_type_cloud_area_fraction	0.16	0.0	-	-
integral_of_surface_downwelling_SW	-0.207	0.004	-	-
y_wind_10m	0.19	0.006	**x**	-
air_temperature_lowest_level	-0.04	0.006	**x**	-
air_pressure_at_sea_level	-0.229	0.011	**x**	-
Hours since model init	0.001	0.0	-	-

B.5 ROC curves

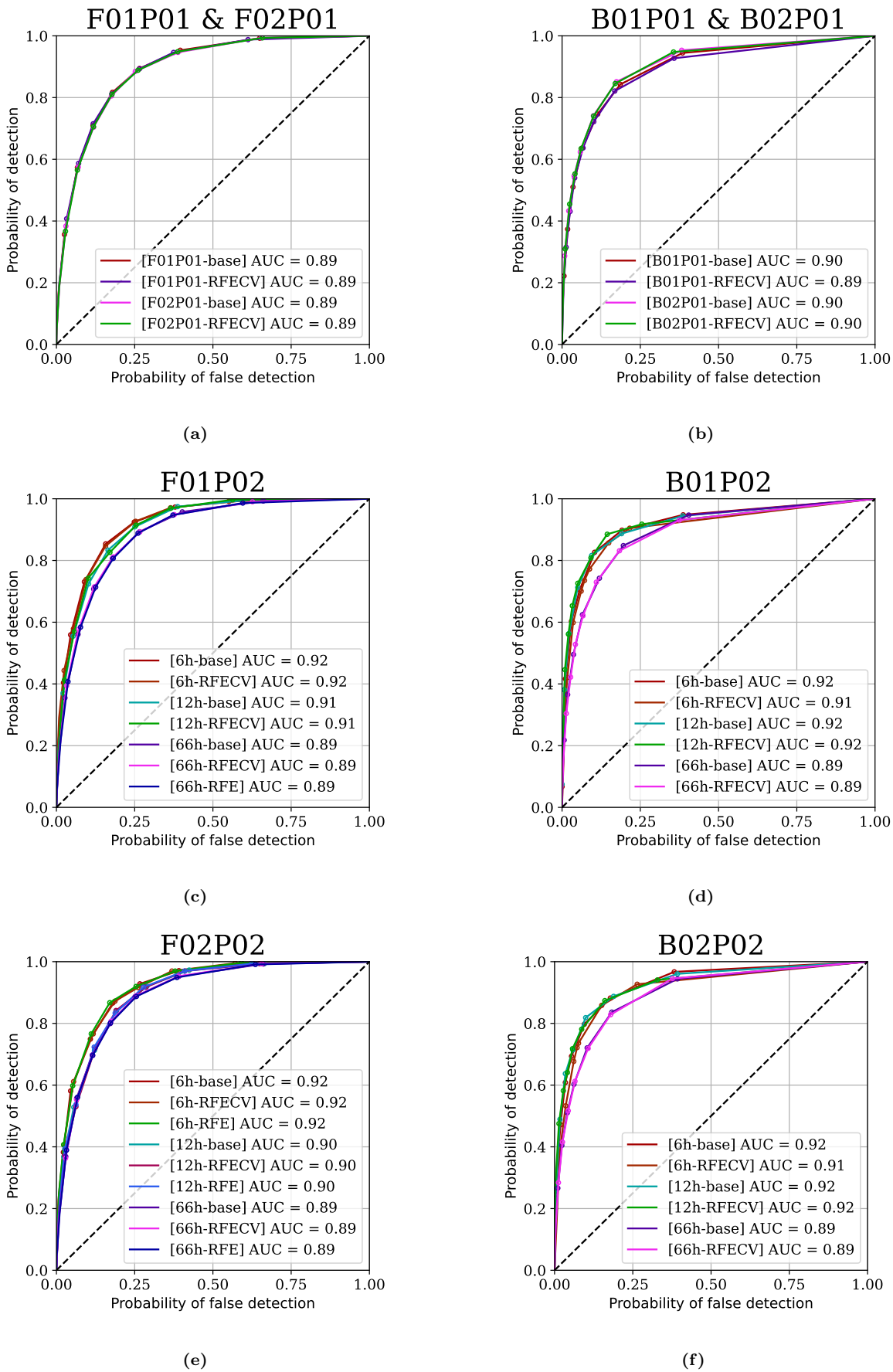


Figure B.5: Figure shows ROC curve for all models trained for feature selection. Left columns shows the ROC curves for Florida, showing F01P01 and F02P01 to the upper left, F01P02 in the middle and F02P02 in the lower left. The equivalent for Blindern is found on the right. The different models refer to the *base* RFC trained on the different feature selections, with *base* referring to all features, and RFECV/RFE refer to the *base* RFC trained on features marked with **x** in tables B.3, B.5, B.4 and B.6.

B.6 Verification metrics

B.6.1 Florida

Table B.8: Table shows the verification metrics for all models run during feature selection for Florida. the upper panel shows the model performance of F01P01 and F02P01, and the following panels shows model performance for P02-6h, P02-12h and P02-66h respectively for both F01 and F02. The different models referred is tha same as explained in fig. B.5.

	[F01P01] Base	[F02P01] Base	[F01P01] RFECV	[F01P01] RFE	-	-	MEPS
ACC	0.84	0.83	0.84	0.83	-	-	0.79
POD	0.65	0.65	0.66	0.66	-	-	0.5
POFD	0.09	0.09	0.09	0.1	-	-	0.09
FAR	0.26	0.27	0.27	0.27	-	-	0.32
SR	0.74	0.73	0.73	0.73	-	-	0.68
CSI	0.53	0.52	0.53	0.53	-	-	0.4
SS	0.42	0.41	0.42	0.41	-	-	0.26
Bias	0.88	0.89	0.9	0.91	-	-	0.73
	[F01P02-6h] Base	**[F02P02-6h] Base**	**[F01P02-6h] RFECV**	**[F01P02-6h] RFE**	**[F02P02] 6h RFECV**	**[F02P02] 6h rfe**	**[6h] MEPS**
ACC	0.85	0.85	0.85	-	0.86	0.86	0.82
POD	0.65	0.67	0.65	-	0.69	0.69	0.54
POFD	0.07	0.07	0.07	-	0.08	0.08	0.07
FAR	0.21	0.22	0.21	-	0.23	0.23	0.24
SR	0.79	0.78	0.79	-	0.77	0.77	0.76
CSI	0.56	0.56	0.55	-	0.57	0.57	0.46
SS	0.48	0.48	0.48	-	0.49	0.49	0.37
Bias	0.83	0.85	0.82		0.9	0.9	0.72
	[F01P02-12h] Base	**[F02P02-12h] Base**	**[F01P02-12h] RFECV**	**[F01P02-12h] RFE**	**[F02P02-12h] RFECV**	**[F02P02-12h] RFE**	**[12h] MEPS**
ACC	0.85	0.83	0.85	-	0.83	0.83	0.8
POD	0.65	0.62	0.67	-	0.64	0.64	0.52
POFD	0.07	0.09	0.08	-	0.09	0.09	0.09
FAR	0.22	0.27	0.23	-	0.27	0.27	0.32
SR	0.78	0.73	0.77	-	0.73	0.73	0.68
CSI	0.55	0.5	0.56	-	0.52	0.52	0.42
SS	0.46	0.39	0.47	-	0.41	0.4	0.28
Bias	0.84	0.85	0.87		0.87	0.88	0.76
	[F01P02-66h] Base	**[F02P02-66h] Base**	**[F01P02-66h] RFECV**	**[F01P02-66h] RFE**	**[F02P02-66h] RFECV**	**[F02P02-66h] RFE**	**[66h] MEPS**
ACC	0.83	0.83	0.83	0.83	0.83	0.83	0.79
POD	0.64	0.63	0.65	0.66	0.63	0.64	0.49
POFD	0.09	0.09	0.1	0.1	0.09	0.09	0.09
FAR	0.27	0.26	0.27	0.28	0.26	0.27	0.31
SR	0.73	0.74	0.73	0.72	0.74	0.73	0.69
CSI	0.52	0.52	0.53	0.53	0.52	0.52	0.4
SS	0.41	0.41	0.41	0.41	0.41	0.41	0.27
Bias	0.88	0.86	0.9	0.92	0.86	0.87	0.71

B.6.2 Blindern

Table B.9: same as for tab. B.8, but for all models run during feature selection for Blindern. the upper panel shows the model performance of B01P01 and B02P01, and the following panels shows model performance for P02-6h, P02-12h and P02-66h respectively for both B01 and B02.

	[B01P01]	[B02P01]	[B01P01] RFECV	[B02P01] RFECV	MEPS
ACC	0.9	0.9	0.9	0.9	0.89
POD	0.44	0.49	0.49	0.51	0.56
POFD	0.03	0.03	0.03	0.03	0.05
FAR	0.26	0.26	0.29	0.28	0.37
SR	0.74	0.74	0.71	0.72	0.63
CSI	0.38	0.42	0.41	0.43	0.42
SS	0.29	0.32	0.29	0.32	0.23
Bias	0.6	0.67	0.68	0.71	0.9
	[B01P02-6h] Base	**[B02P02-6h] Base**	**[B01P02-6h] RFECV**	**[B02P02-6h] RFECV**	**[6h] MEPS**
ACC	0.91	0.91	0.91	0.91	0.91
POD	0.51	0.54	0.62	0.59	0.62
POFD	0.02	0.03	0.04	0.04	0.04
FAR	0.17	0.24	0.27	0.31	0.27
SR	0.83	0.76	0.73	0.69	0.73
CSI	0.46	0.47	0.51	0.47	0.5
SS	0.41	0.38	0.4	0.33	0.38
Bias	0.61	0.71	0.85	0.86	0.85
	[B01P02-12h] Base	**[B02P02-12h] Base**	**[B01P02-12h] RFECV**	**[B02P02-12h] RFECV**	**[12h] MEPS**
ACC	0.91	0.92	0.92	0.92	0.91
POD	0.46	0.59	0.6	0.63	0.65
POFD	0.01	0.03	0.02	0.04	0.05
FAR	0.15	0.21	0.19	0.25	0.31
SR	0.85	0.79	0.81	0.75	0.69
CSI	0.43	0.51	0.53	0.52	0.5
SS	0.38	0.44	0.47	0.42	0.36
Bias	0.54	0.75	0.74	0.84	0.95
	[B01P02-66h] Base	**[B02P02-66h] Base**	**[B01P02-66h] RFECV**	**[B02P02-66h] RFECV**	**[66h] MEPS**
ACC	0.9	0.9	0.9	0.89	0.88
POD	0.43	0.46	0.48	0.47	0.56
POFD	0.03	0.03	0.03	0.03	0.06
FAR	0.26	0.29	0.3	0.3	0.39
SR	0.74	0.71	0.7	0.7	0.61
CSI	0.37	0.38	0.4	0.39	0.41
SS	0.28	0.27	0.28	0.27	0.2
Bias	0.59	0.65	0.69	0.67	0.92

B.6.3 F03 and FB

For curiosity, a dataset with manually selected features from F01 and a dataset combining both F01 and B01 were also tested. The results of this can be seen in table B.10.

Table B.10: Verfication metrics for F03 (left) and FB (middle to right). As F03 is pre-picked features, therefore no RFECV was used, only the verification metrics for the *base* RFC is shown on the left. For FB, both RFECV and RFE, in addition to the base metrics, is shown on the right side of the table.

	[F03P01] Base	MEPS	[FB] Base	[FB] RFECV	[FB] RFE	MEPS
ACC	0.83	0.8	0.85	0.85	0.85	0.82
POD	0.65	0.5	0.44	0.46	0.46	0.48
POFD	0.1	0.09	0.04	0.05	0.05	0.09
FAR	0.27	0.3	0.27	0.27	0.28	0.4
SR	0.73	0.7	0.73	0.73	0.72	0.6
CSI	0.52	0.41	0.38	0.39	0.39	0.36
SS	0.41	0.29	0.28	0.29	0.28	0.16
Bias	0.89	0.72	0.6	0.62	0.64	0.8

C Random forest

The results presented below is from the randomly split data.

C.1 Model selection

C.1.1 Hyper-parameter search

Table C.11: Hyperparameters selected by GridSearchCV for each P01 dataset (random split). *Selected features* refers to the dataset with the RFECV/RFE selected features. In contrast, *all features* refers to the original dataset without any feature selection.

		ccp alpha	class weight	criterion	max depth	n estimators
F01P01	**Selected features**	0.0	balanced_subsample'	gini'	10	250
	All features	0.0	'balanced'	'entropy'	25	250
F02P01	**Selected features**	0.0	'balanced_subsample'	'entropy'	25	750
	All features	0.0	'balanced'	entropy'	25	500
B01P01	**Selected features**	0.0	' balanced'	'gini'	10	250
	All features	0.0	'balanced_subsample'	' entropy'	10	50
B02P01	**Selected features**	0.0	'balanced_subsample'	'entropy'	25	500
	All features	0.0	'balanced_subsample'	'entropy	10	250

Table C.12: Similar as tab. C.11, but the hyperparameters selected by GridSearchCV for all P02 dataset. Also note that for the P02 datasets, the grid-search was only done on the RFECV/RFE-selected features, thus **F01P01-6h** refers to the selected feature of F01P02-6h only, and equivalent for the rest o the P02 datasets.

	ccp alpha	class weight	criterion	max depth	n estimators
F01P02-6h	0.015	balanced	gini	5	50

Continued on next page

Table C.12: Similar as tab. C.11, but the hyperparameters selected by GridSearchCV for all P02 dataset. Also note that for the P02 datasets, the grid-search was only done on the RFECV/RFE-selected features, thus **F01P01-6h** refers to the selected feature of F01P02-6h only, and equivalent for the rest o the P02 datasets. (Continued)

	ccp alpha	**class weight**	**criterion**	**max depth**	**n estimators**
F01P02-12h	0.015	balanced_subsample	gini	10	750
F01P02-66h	0.0	balanced	gini	25	50
F02P02-6h	0.015	balanced_subsample	entropy	5	50
F02P02-12h	0.015	balanced_subsample	entropy	10	500
F02P02-66h	0.0	balanced	entropy	10	50
B01P02-6h	0.0	balanced_subsample	gini	5	50
B01P02-12h	0.015	balanced	entropy	5	500
B01P02-66h	0.0	balanced	entropy	25	50
B02P02-6h	0.015	balanced	gini	5	50
B02P02-12h	0.015	balanced	entropy	5	250
B02P02-66h	0.0	balanced	entropy	10	750

C.1.2 Model selection

Table C.13: Verification metrics for all P01 datasets (random split), evaluated on *validation* data in the model selection. The table is divided into a section for each dataset, with each section containing the verification metrics for the GridSearchCV model with the selected features and all features, as well as *base* RFC with the selected features only. *GridSearchCV* here reference the random forest classifier (RFC) with the hyperparameters (as seen in tab. C.11 and C.12) found in the grid-search, while *Base* refers to the *base* RFC used in feature selection (settings specified in tab. 6 in sec. 6.1.2)

		ACC	**POD**	**POFD**	**FAR**	**SR**	**CSI**	**SS**	**Bias**
F01P01	**GridSearchCV**	0.81	0.86	0.21	0.38	0.62	0.56	0.33	1.39
	GridSearchCV (All features)	0.81	0.86	0.21	0.38	0.62	0.57	0.34	1.39
	Base	0.83	0.66	0.1	0.27	0.73	0.53	0.41	0.9
	MEPS	0.8	0.5	0.09	0.3	0.7	0.41	0.29	0.72
F02P01	**GridSearchCV**	0.81	0.86	0.2	0.38	0.62	0.56	0.34	1.38
	GridSearchCV (All features)	0.82	0.86	0.2	0.37	0.63	0.57	0.34	1.37
	Base	0.83	0.65	0.1	0.28	0.72	0.52	0.4	0.9
	MEPS	0.79	0.5	0.09	0.32	0.68	0.4	0.26	0.73
B01P01	**GridSearchCV**	0.84	0.83	0.16	0.54	0.46	0.42	−0.16	1.81
	GridSearchCV (All features)	0.83	0.84	0.17	0.55	0.45	0.42	−0.17	1.86
	Base	0.9	0.48	0.03	0.28	0.72	0.4	0.29	0.67
	MEPS	0.89	0.56	0.05	0.37	0.63	0.42	0.23	0.9
B02P01	**GridSearchCV**	0.84	0.85	0.16	0.52	0.48	0.44	−0.08	1.78
	GridSearchCV (All features)	0.85	0.85	0.15	0.52	0.48	0.44	−0.07	1.77
	Base	0.9	0.51	0.03	0.28	0.72	0.43	0.32	0.71
	MEPS	0.89	0.58	0.06	0.36	0.64	0.44	0.25	0.91

Table C.14: Same as tab. C.13, but for the model selection of all P02 datasets, evaluated on the *validation* data.

		ACC	POD	POFD	FAR	SR	CSI	SS	Bias
F01P02-6h	**GridSearchCV**	0.82	0.88	0.2	0.38	0.62	0.57	0.35	1.4
	Base	0.85	0.66	0.07	0.21	0.79	0.56	0.48	0.84
	MEPS	0.82	0.54	0.07	0.24	0.76	0.46	0.37	0.72
F01P02-12h	**GridSearchCV**	0.82	0.86	0.2	0.36	0.64	0.58	0.37	1.35
	Base	0.85	0.65	0.07	0.22	0.78	0.55	0.46	0.83
	MEPS	0.81	0.54	0.08	0.27	0.73	0.45	0.34	0.74
F01P02-66h	**GridSearchCV**	0.8	0.86	0.22	0.39	0.61	0.55	0.31	1.4
	Base	0.83	0.65	0.1	0.28	0.72	0.52	0.4	0.91
	MEPS	0.79	0.5	0.09	0.32	0.68	0.4	0.27	0.73
F02P02-6h	**GridSearchCV**	0.82	0.88	0.2	0.37	0.63	0.58	0.37	1.4
	Base	0.86	0.69	0.07	0.22	0.78	0.58	0.5	0.88
	MEPS	0.83	0.57	0.07	0.24	0.76	0.48	0.39	0.74
F02P02-12h	**GridSearchCV**	0.81	0.87	0.22	0.39	0.61	0.56	0.31	1.43
	Base	0.84	0.66	0.09	0.27	0.73	0.53	0.42	0.91
	MEPS	0.8	0.52	0.09	0.32	0.68	0.42	0.28	0.76
F02P02-66h	**GridSearchCV**	0.81	0.86	0.21	0.38	0.62	0.56	0.33	1.38
	Base	0.83	0.64	0.09	0.27	0.73	0.52	0.4	0.88
	MEPS	0.79	0.49	0.09	0.31	0.69	0.4	0.27	0.71
B01P02-6h	**GridSearchCV**	0.86	0.85	0.14	0.49	0.51	0.47	0.04	1.66
	Base	0.91	0.62	0.04	0.27	0.73	0.51	0.4	0.85
	MEPS	0.91	0.63	0.04	0.26	0.74	0.52	0.41	0.86
B01P02-12h	**GridSearchCV**	0.88	0.86	0.12	0.45	0.55	0.5	0.15	1.57
	Base	0.92	0.6	0.02	0.19	0.81	0.53	0.47	0.74
	MEPS	0.92	0.67	0.04	0.26	0.74	0.54	0.43	0.91
B01P02-66h	**GridSearchCV**	0.83	0.83	0.17	0.55	0.45	0.41	-0.2	1.86
	Base	0.9	0.48	0.04	0.3	0.7	0.4	0.28	0.69
	MEPS	0.88	0.56	0.06	0.39	0.61	0.41	0.2	0.93
B02P02-6h	**GridSearchCV**	0.87	0.84	0.12	0.48	0.52	0.48	0.08	1.6
	Base	0.91	0.59	0.04	0.31	0.69	0.47	0.33	0.86
	MEPS	0.91	0.62	0.04	0.27	0.73	0.5	0.38	0.85

Continued on next page

Table C.14: Same as tab. C.13, but for the model selection of all P02 datasets, evaluated on the *validation* data. (Continued)

		ACC	POD	POFD	FAR	SR	CSI	SS	Bias
B02P02-12h	**GridSearchCV**	0.86	0.86	0.14	0.49	0.51	0.47	0.04	1.67
	Base	0.92	0.63	0.04	0.25	0.75	0.52	0.42	0.85
	MEPS	0.91	0.65	0.05	0.31	0.69	0.5	0.36	0.95
B02P02-66h	**GridSearchCV**	0.83	0.84	0.17	0.55	0.45	0.42	-0.18	1.85
	Base	0.9	0.48	0.03	0.3	0.7	0.39	0.27	0.68
	MEPS	0.88	0.56	0.06	0.39	0.61	0.41	0.2	0.92

C.1.3 Model evaluation

Table C.15: Verification metrics for the final random forest classifier for all datasets (random split) and MEPS, evaluated on the previously *unseen test* data.

	ACC	POD	POFD	FAR	SR	CSI	SS	Bias
F01P01	0.83	0.65	0.1	0.27	0.73	0.52	0.4	0.89
F02P01	0.83	0.65	0.09	0.27	0.73	0.53	0.41	0.89
MEPS	0.8	0.51	0.09	0.31	0.69	0.41	0.28	0.73
F01P02-6h	0.84	0.63	0.07	0.23	0.77	0.53	0.44	0.82
F02P02-6h	0.84	0.65	0.08	0.24	0.76	0.54	0.44	0.86
MEPS	0.81	0.53	0.08	0.27	0.73	0.44	0.33	0.72
F01P02-12h	0.84	0.66	0.09	0.26	0.74	0.53	0.42	0.89
F02P02-12h	0.86	0.7	0.08	0.24	0.76	0.57	0.47	0.92
MEPS	0.8	0.52	0.09	0.31	0.69	0.43	0.29	0.76
F01P02-66h	0.83	0.65	0.1	0.28	0.72	0.52	0.39	0.91
F02P02-66h	0.83	0.64	0.09	0.27	0.73	0.52	0.4	0.88
MEPS	0.79	0.49	0.09	0.32	0.68	0.4	0.26	0.73
B01P01	0.9	0.5	0.03	0.29	0.71	0.42	0.3	0.7
B02P01	0.9	0.49	0.03	0.29	0.71	0.41	0.29	0.69
MEPS	0.89	0.58	0.06	0.37	0.63	0.43	0.24	0.92
B01P02-6h	0.9	0.54	0.04	0.3	0.7	0.44	0.31	0.78
B02P02-6h	0.91	0.61	0.04	0.28	0.72	0.49	0.37	0.84
MEPS	0.9	0.58	0.04	0.28	0.72	0.48	0.36	0.81
B01P02-12h	0.91	0.58	0.03	0.24	0.76	0.49	0.4	0.76
B02P02-12h	0.91	0.62	0.04	0.26	0.74	0.51	0.4	0.83
MEPS	0.9	0.63	0.05	0.3	0.7	0.5	0.36	0.91
B01P02-66h	0.9	0.47	0.03	0.31	0.69	0.39	0.26	0.68
B02P02-66h	0.9	0.49	0.03	0.29	0.71	0.41	0.29	0.68
MEPS	0.89	0.57	0.06	0.39	0.61	0.42	0.21	0.93

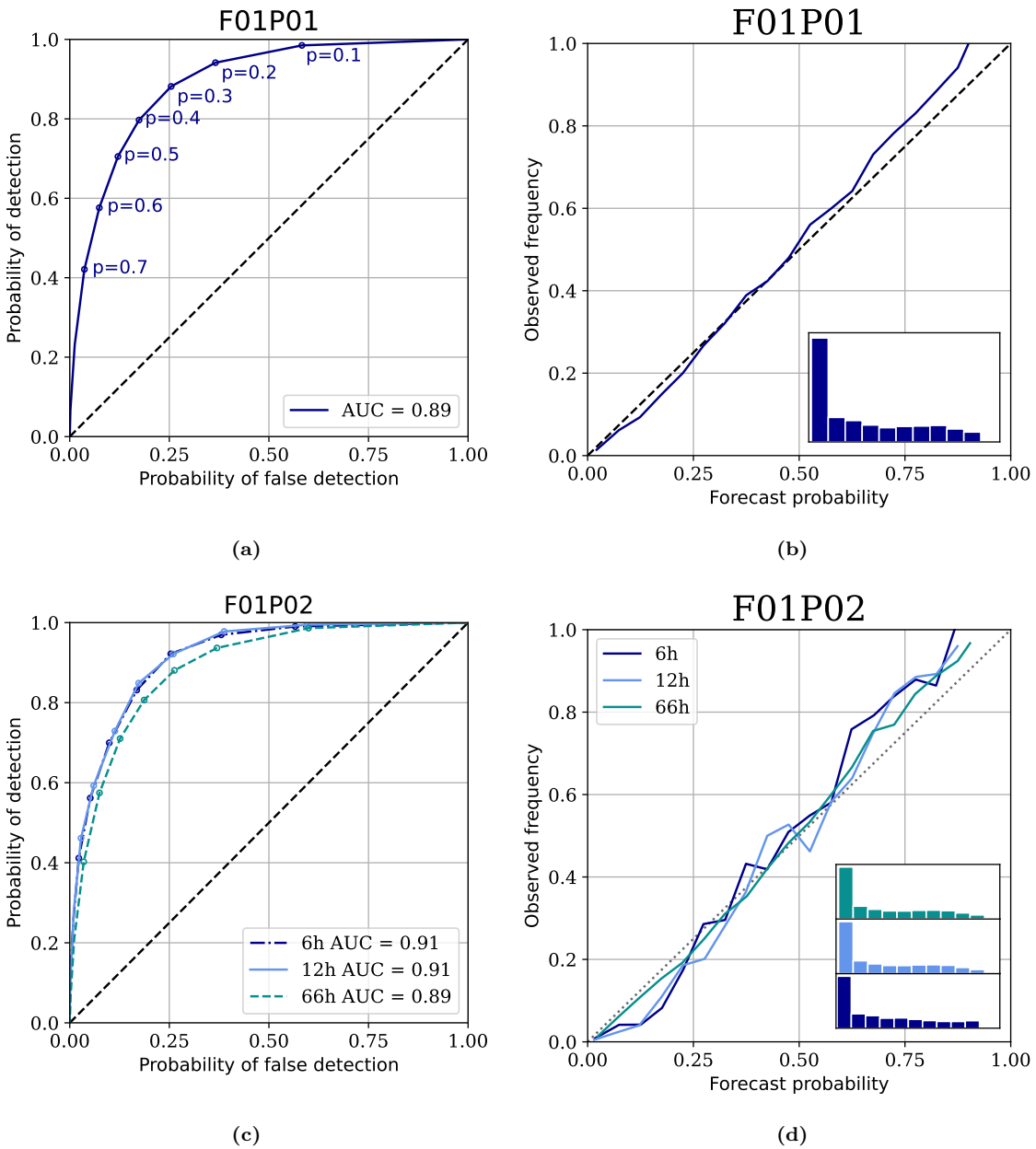


Figure C.6: Left figures shows the the final (i.e base) RFC trained on the RFECV/RFE selected features for F01P01 (upper) and F02P01 (lower). Left figures show the ROC curve, while right figures show the reliability diagram.

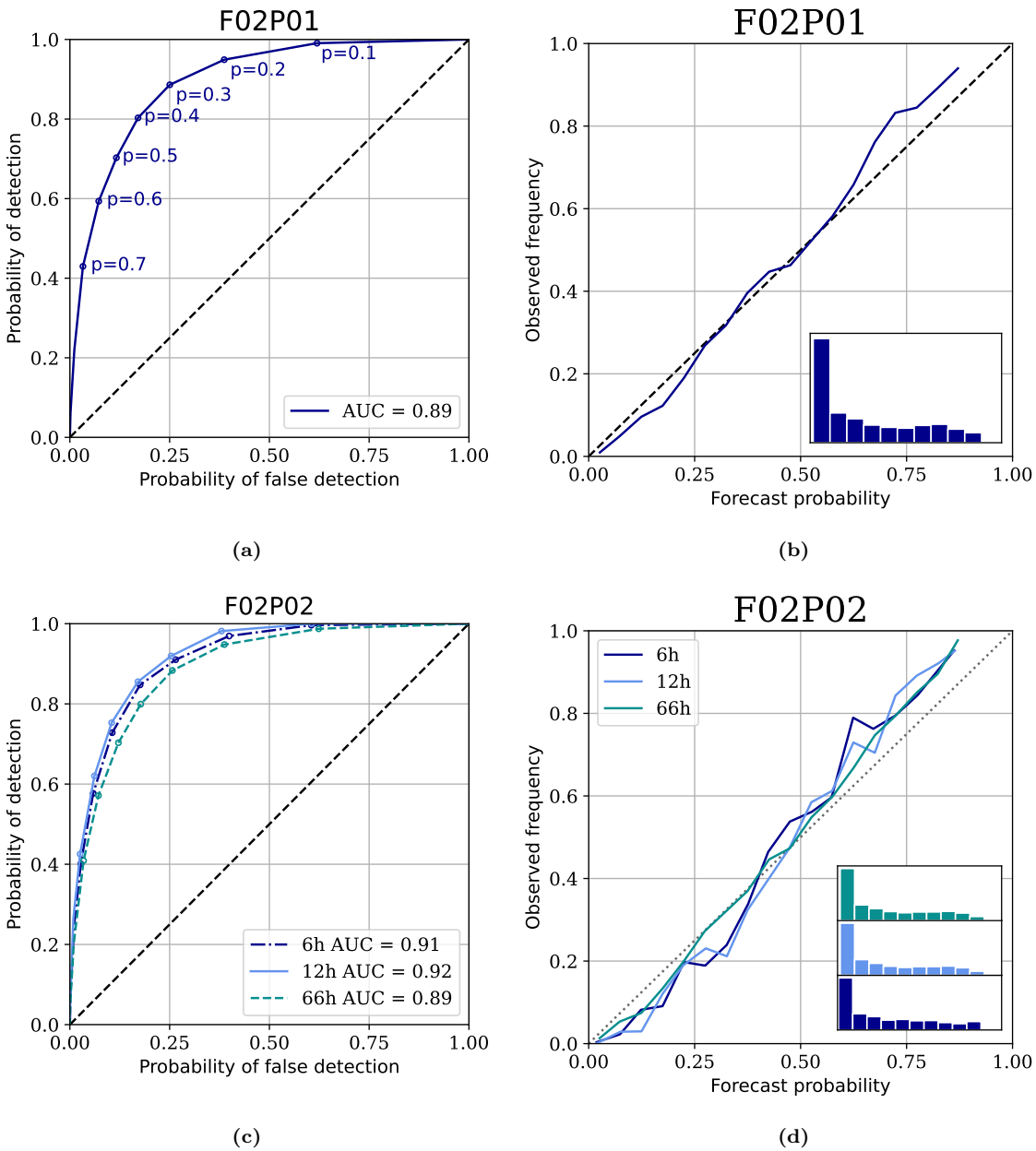


Figure C.7: Same as fig. C.6, but for F02P01 and F02P02.

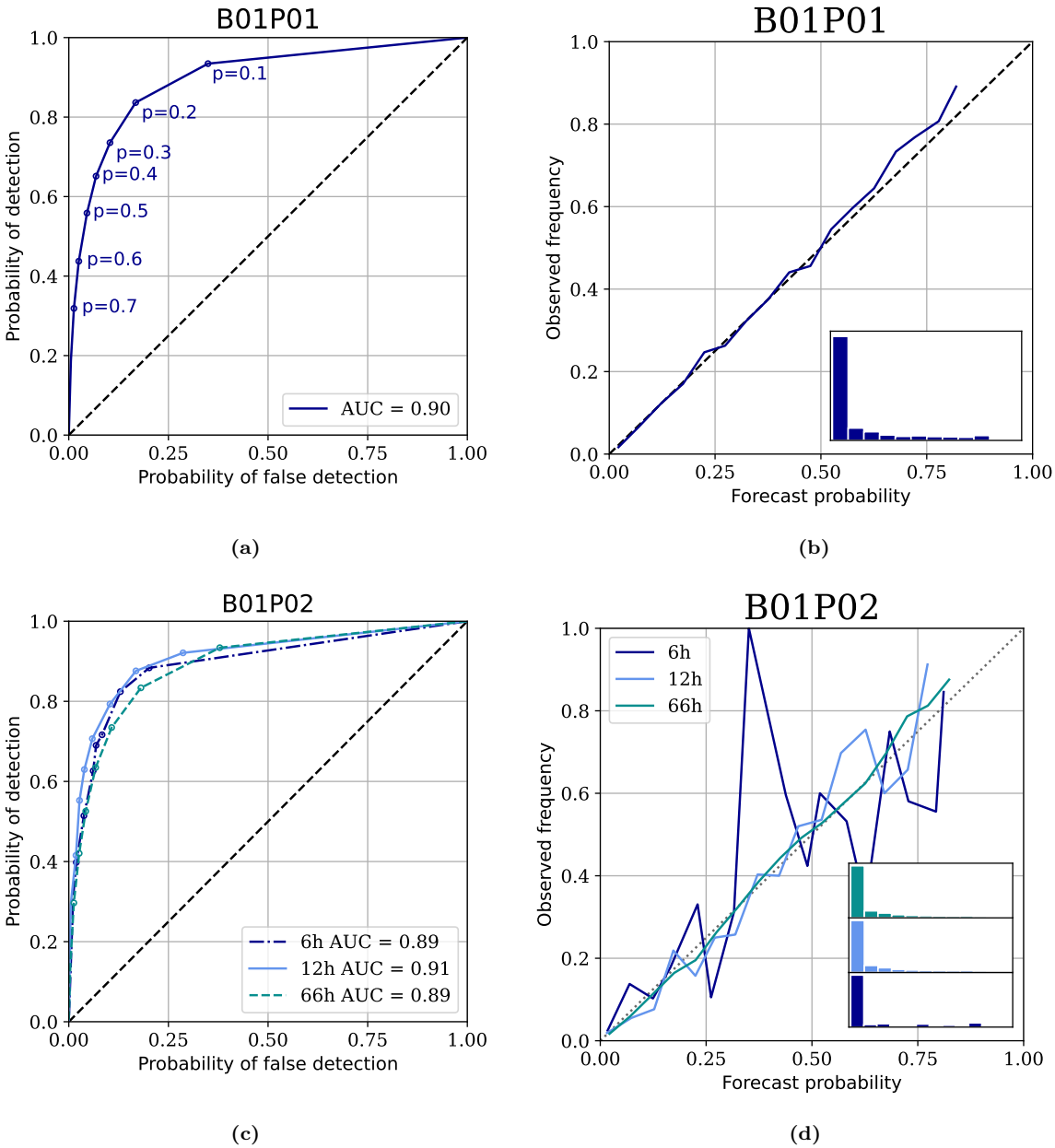


Figure C.8: Same as C.6, but for B01P01 and B01P02.

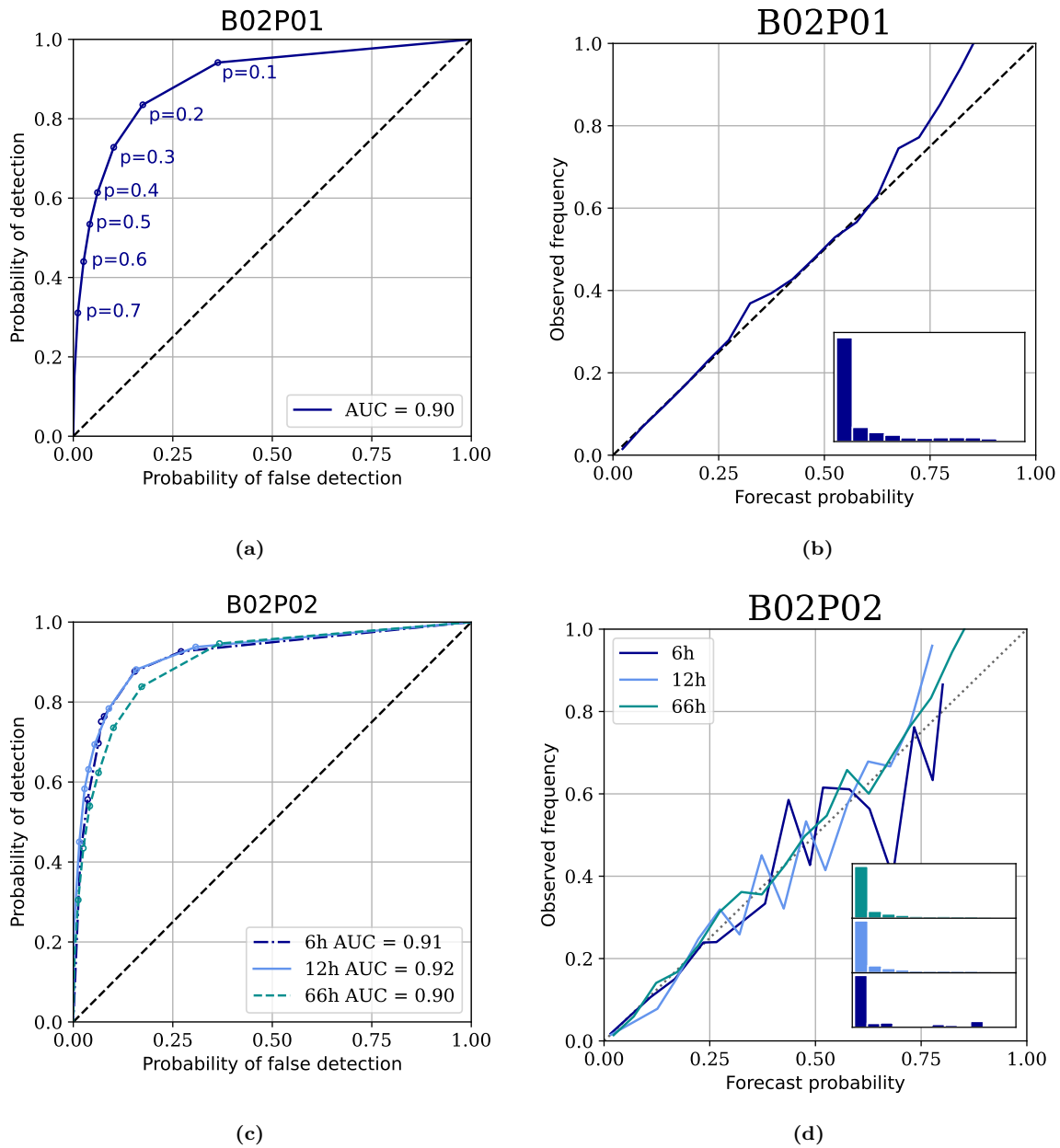


Figure C.9: Same as C.6, but for B02P01 and B02P02.

D Neural Network

D.1 NN trained on random split predict 2022 precipitation

Table D.16: The MAE for the NN trained on randomly drawn data compared to MEPS. The MAE is caculated from the total forecast (i.e combined class 0 (*no precipitation*) and class 1 (*precipitation*)). The top rows show the MAE on **randomly drawn** *test* data, and bottom rows show 2022 data between January and August. The evaluation done with randomly drawn test data show significantly more optimistic results (i.e greater rduction in MAE compared to MEPS) than the evaluation on 2022 data. This illustrated how auto-correlation within the data provide a biased evalution when split randomly into training and test data.

		MAE					
		All rates	**No rain**	**Light rain**	**Moderate rain**	**Heavy rain**	
Test data	NN	0.228	0.029	0.525	2.052	4.758	
	MEPS	0.299	0.062	0.65	2.468	5.844	
Jan-Jul 2022	NN	0.413	0.157	0.647	3.408	8.608	
	MEPS	0.459	0.211	0.691	3.349	8.628	

Table D.17: Similarly to tab. D.16, for each induvidual class seperatly (i.e class 0: *precipitation*, class 1: *precipitation*), all all ranges. The evaluation done on 2022 data is presented in the left column, and the evaluation done on randomly drawn *test* data is shown n the right column.

		MAE	
		Jan - Aug 2022	**Test data**
Class 0	**NN**	0.221	0.124
	MEPS	0.215	0.155
Class 1	**NN**	0.75	0.534
	MEPS	0.408	0.761

D.2 Base NN and feature selection

Table D.18: Table show an overview of the evaluation of the *base* neural network (NN) (evaluated on *validation* data). The three columns show the *base* NN trained on the features selected by the recursive feature elimination (RFE) presented in sec. 11.2 (left), the RFE-selected features with additional tendency features (middle) and MEPS (right). The tendency features included are marked in gray in tab. 4 in sec. 5. Note that for B01P01, the process of model selection and final model evaluation was done on both the RFE-only features and the RFE+tendnecy.

			MAE		
			RFE vars only	**Tendency vars included**	**MEPS**
F01P01	**Base NN**	Class 0	0.165	0.157	0.600
		Class 1	0.666	0.635	0.976
F02P01	**Base NN**	Class 0	0.123	0.125	0.800
		Class 1	0.726	0.709	1.147
B01P01	**Base NN**	Class 0	0.074	0.081	0.052
		Class 1	0.529	0.585	0.604
	opt NN	Class 0	0.049	0.049	0.051
		Class 1	0.503	0.463	0.604
B02P01	**Base NN**	Class 0	0.078	0.069	0.047
		Class 1	0.513	0.534	0.608

D.3 Model selection and evaluation: F01P01 - class 1

This section present the optimal neural network found in a grid-search for F01P01.

Table D.19: Table show the archetecture of the optimized neural network found for F01P01 from a grid-search. Only *one* hidden layer was selected, with 128 neurons in both the *input* layer and the *hidden* layer.

Architecture of the network		
	Class 1	
F01P01	**Layer**	**Units**
	Input	128
	Hidden 1	128
	Output	1
	Activation	*ReLU*

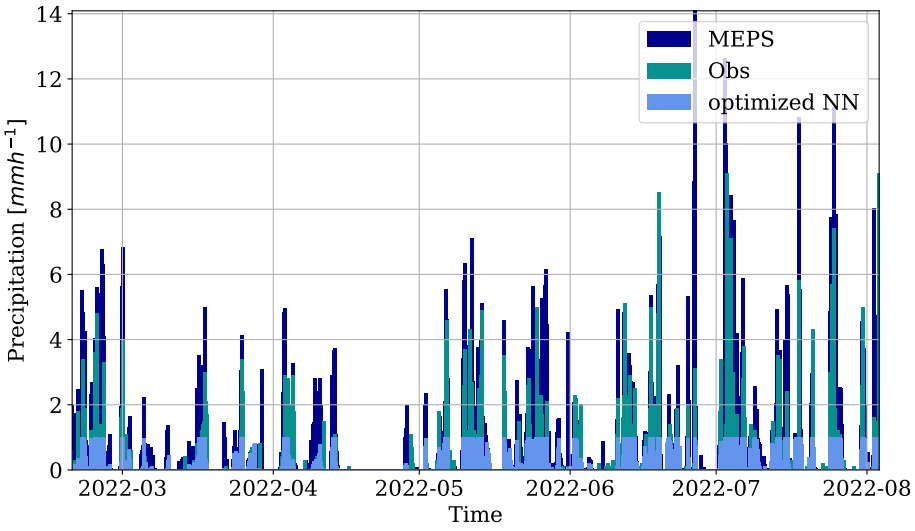


Figure D.10: Figure shows the time series for the the predictions from both the 1-hidden layer NN and MEPS, together with the corresponding observations, for the *test* data spanning 18 February to 3 August 2022. It is clear that the neural network is unable to capture the variability seen in the precipitation, and that there is an artificial cut-off. The verification metrics of the 1-hidden layer NN is presented in tab. D.20.

D.3.1 Model evaluation

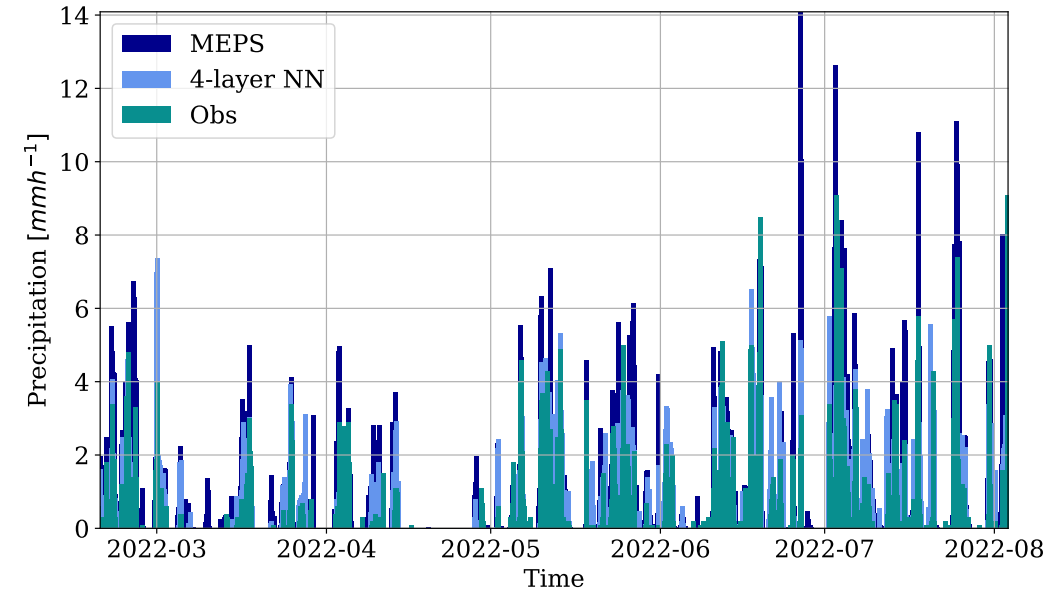
Table D.20: Verification metrics for the optimized neural network trained for F01P01 (upper half) and B01P01 (lower half). Class 0 refers to the neural network (NN) trained on data-points classified as *no precipitation* by the RFC, while *class 1* refers to the NN trained on those classified as *precipitation*. The upper panel, referred to as *total* is the overall mean absolute error (MAE) and bias. The columns shows the MAE and bias for the different ranges of precipitation rate. The middel panel show the corresponding verification metrics for class 0 (*no precipitation*), while the lower show the same for class 1 (*precipitation*).

				All rates	No rain	Light rain	Moderate rain	Heavy rain
F01P01	Total	MAE	NN (1hidden layer)	0.178	0.027	0.48	3.089	8.186
			NN (4hidden layers)	0.205	0.042	0.587	2.768	7.986
			MEPS	0.223	0.061	0.627	2.553	7.349
		Bias	NN (1hidden layer)	-0.108	0.027	-0.339	-3.089	-8.186
			NN (4hidden layers)	-0.068	0.042	-0.231	-2.723	-7.986
			MEPS	-0.009	0.061	-0.042	-2.324	-7.349
		SS	NN (1hidden layer)	0.099	-inf	0.236	0.168	0.057
			NN (4hidden layers)	-0.035	-inf	0.067	0.255	0.08
			MEPS	-0.126	-inf	0.002	0.313	0.153
	Class 0	MAE	NN	0.047	0.0	0.428	3.656	8.775
			MEPS	0.056	0.012	0.41	3.525	8.734
		Bias	NN			-0.428	-3.656	-8.775
			MEPS	-0.029	0.012	-0.353	-3.525	-8.734
		SS	NN	0.0		0.0	0.0	0.0
			MEPS	-0.201	-inf	0.043	0.036	0.005
	Class 1	MAE	NN (1hidden layer)	0.602	0.239	0.506	3.017	8.108
			NN (4hidden layers)	0.713	0.372	0.663	2.655	7.88
			MEPS	0.759	0.44	0.732	2.429	7.164
		Bias	NN (1hidden layer)	-0.306	0.239	-0.295	-3.017	-8.108
			NN (4hidden layers)	-0.138	0.372	-0.136	-2.604	-7.88
			MEPS	0.054	0.44	0.109	-2.17	-7.164
		SS	NN (1hidden layer)	0.12	-inf	0.303	0.19	0.064
			NN (4hidden layers)	-0.042	-inf	0.086	0.287	0.091
			MEPS	-0.109	-inf	-0.009	0.348	0.173

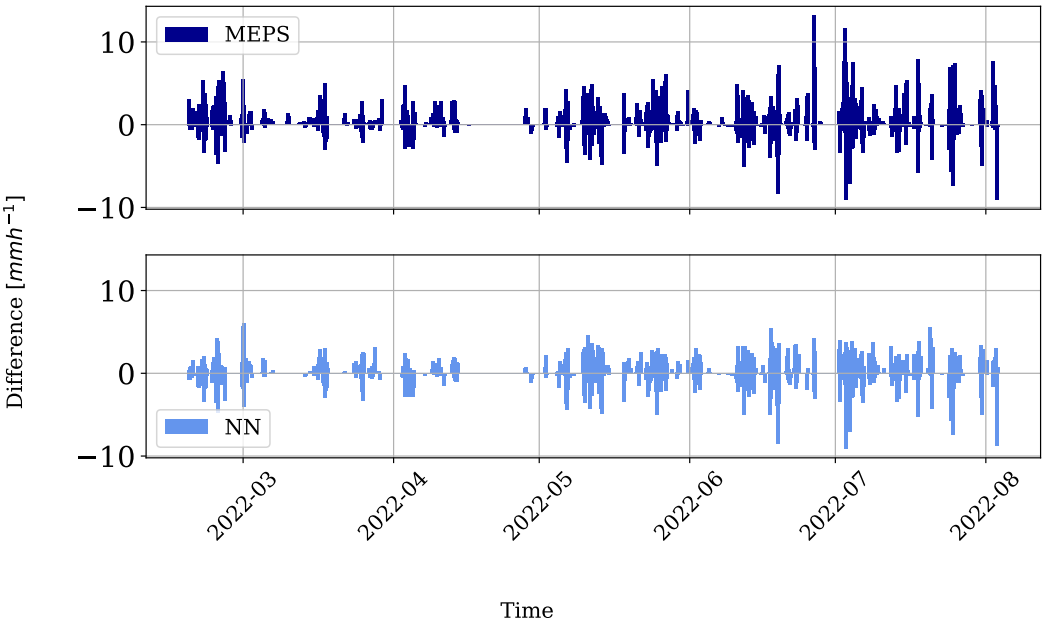
Continued on next page

Table D.20: Verification metrics for the optimized neural network trained for F01P01 (upper half) and B01P01 (lower half). Class 0 refers to the neural network (NN) trained on data-points classified as *no precipitation* by the RFC, while *class 1* refers to the NN trained on those classified as *precipitation*. The upper panel, referred to as *total* is the overall mean absolute error (MAE) and bias. The columns shows the MAE and bias for the different ranges of precipitation rate. The middel panel show the corresponding verification metrics for class 0 (*no precipitation*), while the lower show the same for class 1 (*precipitation*). (Continued)

				All rates	**No rain**	**Light rain**	**Moderate rain**	**Heavy rain**
B01P01	**Total**	**MAE**	NN	0.065	0.006	0.493	3.339	10.547
			MEPS	0.097	0.035	0.609	2.745	10.382
		Bias	NN	-0.05	0.006	-0.44	-3.339	-10.547
			MEPS	0.009	0.035	-0.062	-2.481	-9.79
		SS	NN	0.034	-inf	0.148	0.078	0.01
			MEPS	-0.433	-inf	-0.052	0.242	0.025
	Class 0	**MAE**	NN	0.039	0.0	0.487	3.834	11.576
			MEPS	0.048	0.01	0.495	3.67	11.462
		Bias	NN	-0.039	0.0	-0.487	-3.834	-11.576
			MEPS	-0.023	0.01	-0.362	-3.67	-11.462
		SS	NN	0.0		0.0	0.0	0.0
			MEPS	-0.243	-inf	-0.017	0.043	0.01
	Class 1	**MAE**	NN	0.56	0.26	0.505	2.907	7.046
			MEPS	1.004	1.082	0.809	1.935	6.71
		Bias	NN	-0.265	0.26	-0.357	-2.907	-7.046
			MEPS	0.599	1.082	0.462	-1.441	-4.104
		SS	NN	0.075	-inf	0.319	0.154	0.061
			MEPS	-0.66	-inf	-0.092	0.437	0.105

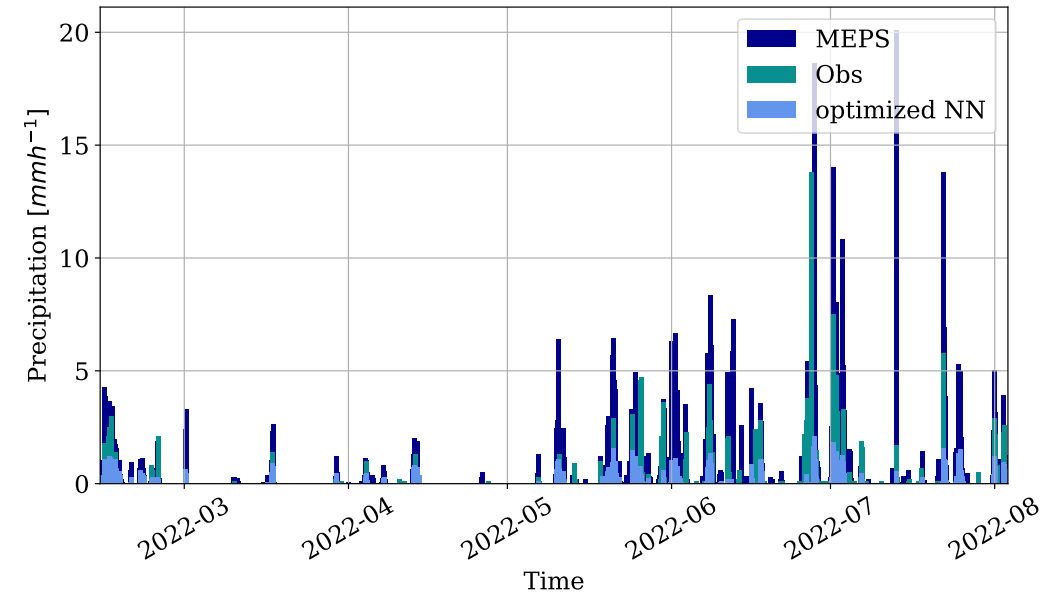


(a)

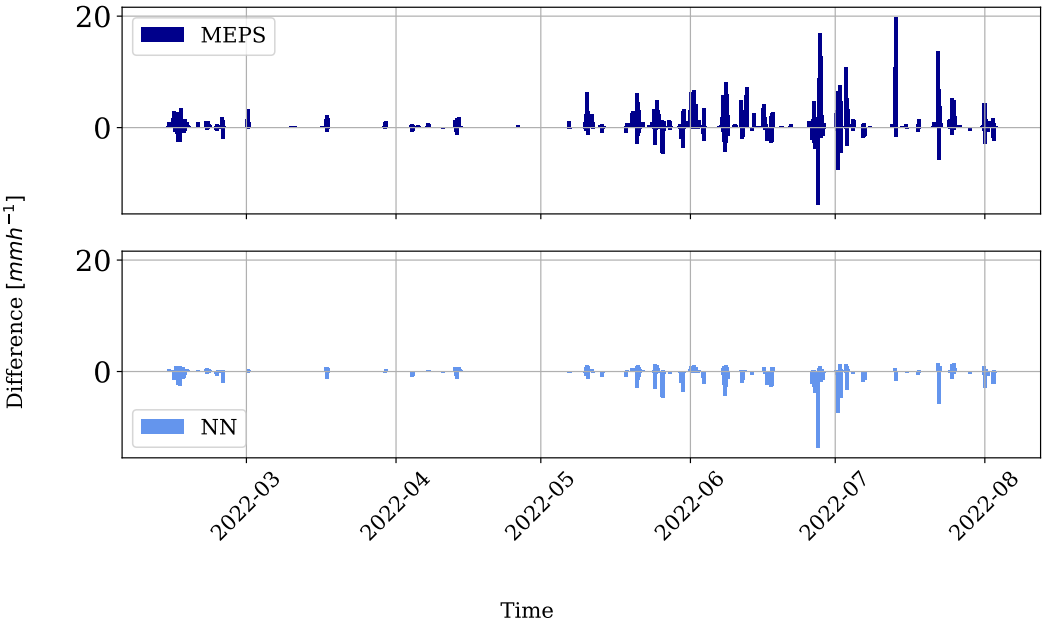


(b)

Figure D.11: The figure shows a time-series plot for the model prediction for the verification data (i.e the *test* data) from MEPS (navy), the neural network (light blue) and the observations (teal) for (D.11a) Florida (F01P01). The difference between the forecast and observation ($forecast - observation$) is shown in D.11b.



(a)



(b)

Figure D.12: Same as for fig. D.11, but for Blindern (B01P01). Fig. D.12a show the time-series of the model predictions, and fig. (D.12b) show the difference between the forecast and observation ($forecast - observation$).