Winter Wheat Harvest Prediction Using Primarily Satellite Radar Data from Sentinel-1

Oliver Persson Bogdanovski, Christoffer Svenningsson

Master's thesis
2022:E74

LUND UNIVERSITY

Faculty of Engineering
Centre for Mathematical Sciences
Mathematics

CENTRUM SCIENTIARUM MATHEMATICARUM

LUND UNIVERSITY

LTH

FACULTY OF ENGINEERING

Master's thesis

Winter Wheat Harvest Prediction Using Primarily Satellite Radar Data from Sentinel-1

authors

Oliver Persson Bogdanovski

Christoffer Svenningsson

supervisor

Alexandros Sopasakis, Docent

examiner

Matthias Ohlsson, Professor

Tue 10th Jan, 2023

Abstract

Aiding farmers with their tremendous task of sustainably and cost-efficiently feeding the world is of utmost importance. Information technology plays a crucial role in supporting farmers and supplying them with accurate information about their crops. The information gives farmers operative benefits, such as optimized timing of fertilization and plant protection to name a few.

Satellite optical imagery from Sentinel-2 satellites has been used to predict harvest but is severely hampered by cloud cover, which is not the case for Synthetic Aperture Radar (SAR) backscatter from Sentinel-1 satellites. In our thesis, we used primarily Sentinel-1 satellite data and additional weather, topography, and elevation data to predict the winter wheat harvest on selected fields in Skåne, a region in Southern Sweden. We compared the performance of two machine learning models: Light Gradient-Boosting Machines (LGBM) and Feedforward Neural Networks.

Our results show that Sentinel-1 data contains valuable information for winter wheat harvest prediction, and can achieve a top RMSE of 0.74 tonnes per hectare using all data and LGBM. We tested different resolutions of the harvest data grid. To our surprise, the lower-resolution grid outperforms the higher-resolution grid. Furthermore, we tested if transfer learning between years were possible. In general, we could not achieve transfer learning, as the harvest data distribution varied greatly for each season.

Further work is needed to investigate why the lower-resolution grid outperformed the higher-resolution grid and to model the varying harvest distribution. Moreover, the combination of both Sentinel-1 and Sentinel-2 data might lead to better results, since it is possible that Sentinel-1 backscatter contains information that can not be derived from Sentinel-2 optical imagery and vice versa.

Keywords: Precision Agriculture, Sentinel-1 SAR, Machine Learning, Winter Wheat, Harvest Prediction, RFI-filtering, Despeckling

Acknowledgment

Thank you, Alexandros, for your irreplaceable work as our supervisor and mentor. Among the talented and devoted teachers at Lund University, you stand out and will occupy a special place in our hearts. We would also like to give a special thank you to Henrik for providing us with the essential harvest data and for persevering when the technology presented challenges in our need for a finer grid. Another greatly valuable person to our thesis was you, Simon. Thank you for helping us with the weather, topology, and elevation data. Your work saved us a lot of work and we would not have been able to do it without you.

Contents

Chapter 1

Introduction

Aiding farmers with their tremendous task of feeding the world is of utmost importance. To feed all of mankind we need more food, whilst at the same time, we need to preserve our ecosystems. To produce more food without any new innovation, farmers need to cultivate larger and larger areas of land, which comes at the expense of the planet's ecosystems and their wallets. Information technology can aid in this issue. Supplying accurate information to farmers may lead to an increase in efficiency. With higher efficiency, the farmers can achieve larger harvests, potentially using fewer resources and lowering the costs of their operations. This may free up land for preservation and make food more affordable - helping both the Earth and mankind.

In modern-day society machine learning (ML) and data are omnipresent, and can be used for different ends. There are several positive applications of ML, for example predicting protein folding [5], self-driving cars [2], and cancer prognosis [25] to name a few. ML can, however, also be used for morally questionable activities, such as surveillance of citizens [51]. Data can be used to build vast fortunes by financial modeling of the stock market [18] or better performing ads fueling consumption [14]. We believe that with great power comes great responsibility[1] and that the great power gained by ML and data should be used for meaningful ends.

We will investigate the possibilities of using ML models to predict winter wheat harvest on fields in the region of Skåne, Sweden. Why is this important? To put it simply, with accurate information farmers can make well-informed decisions for effective operations [36]. With ML models, a farmer might be able to use fewer pesticides and less fertilizer and still be able to achieve a larger harvest than without it. With ML models, a farmer might be able to achieve a larger harvest using less land, leading to less energy used by machinery and freeing up land for

[1] Made famous by the great comic author Stanley Lieber.

the regrowth of natural habitat [24].

Now that we understand *what* we need and *why*, *how* do we create the ML models? We need the aforementioned - data. A multitude of data will be used to train the ML models, with the primary data coming from Sentinel-1 backscatter and harvest data from the fields. Our thesis will leverage ML and data to build models which, hopefully, will be able to accurately help farmers with their operations. The concept of using ML and data to help farmers is part of *precision agriculture*.

1.1 Goals & Research Questions

The main goal of this thesis was to investigate the possibility of using radar backscatter from Sentinel-1 satellites for harvest prediction on winter wheat fields in Skåne, a region in Southern Sweden. To be able to reach the main goal, several sub-goals were set:

- Pre-processing of the Sentinel-1, weather, topology and elevation data
- Georeferencing satellite data with local harvest data from fields
- Extensive coding to facilitate the thesis, e.g. downloading satellite data, running evaluation of models, etc.

From the main goal, several research questions were sparked, such as:

- How do different ML models affect performance?
- What is the effect of computed indices of satellite bands?
- How do weather data, topology classification, and elevation data affect the performance?
- Are models trained with data from one year able to predict another year's harvest?

1.2 Division of Work

In general, the coding was carried out in tandem, frequently pair programming. If not stated otherwise, equal collaborative effort can be assumed with all parts of the code. We had some separation of responsibility, with Christoffer focusing on despeckling of Sentinel-1 data, radio frequency interference filtering, weather data handling, topology data, and elevation data whilst Oliver focused on data set

creation, model building, and experiments. However, individual responsibilities should not imply complete isolation, as we frequently offered a lending hand when individual progress haltered.

In general, the report was written as a team with all parts being thoroughly discussed and reviewed. If not stated otherwise, equal collaborative effort can be assumed with all parts of the report. Some separation of the writing process was done. Christoffer had mainly the responsibility of writing about precision agriculture in the Background. For the Methodology, he was the main author of despeckling, radio frequency interference filtering, light gradient boosting machines, and transfer learning. Further, he was the main writer of all parts of the Discussion and Conclusion. Oliver was the main writer responsible for writing the Abstract and Introduction. In the Background, he had the main responsibility of Sentinel-1, machine learning, related work, and scientific contribution. In the Methodology, his main responsibility was Sentinel-1 data (excluding despeckling and radio metric interference), sampling, feedforward neural networks, metrics, and k-fold cross-validation. The Result section was mainly his responsibility, providing all figures and tables and writing most of the text. Similarly to the coding, individual responsibilities should not imply complete isolation, as we frequently offered a lending hand to review and discuss the content of each other's sections.

Chapter 2

Background

In this chapter, we will introduce several topics relevant to the thesis. We will start by presenting precision agriculture and then a more thorough presentation of Sentinel-1. We will also mention briefly machine learning, free and open-source software, and Alvis. Finally, we will discuss related work and our scientific contribution.

2.1 Precision Agriculture

Precision agriculture is the practice of using science and technology to improve farming in order to achieve the best outcome possible [27]. More precisely, this means that we want to maximize yield while sustaining the long-term health of the soil and having the smallest possible impact on the environment. This is done by using technology to collect information about the local variations within fields to adapt the use of fertilizers, seeds, crop nutrients, plant protection, etc. [48].

Most of the modern precision agriculture practices can be divided into the following three phases [30].

1. Data collection

2. Data analysis

3. Operational application

In the first phase of data collection, a variety of methods can be employed depending on the specific application. Some common techniques include taking soil samples, using unmanned aerial vehicles (UAVs) equipped with remote sensing devices, and using modern farming machinery such as combine harvesters to collect data about local variations in yield.

The second phase, data analysis commonly consists of employing a mathematical model, e.g. a machine learning model, to understand what the data says about the field. Our work mainly contributes to this phase of precision agriculture.

The third phase, the operational application consists of using the obtained knowledge about the field in order to improve the yield, minimize the environmental impact, or simply save money by only using products such as fertilizer where it is needed.

2.1.1 Harvest Data

For this project, we have received harvest data for winter wheat fields in Skåne. The data was collected with combine harvesters[1] in the years 2017, 2018, 2019, and 2020 during the harvest season. The data was preprocessed and resampled into a format suitable for machine learning before being sent to us. The specifics of the harvest data are described later (Section 3.2).

2.2 The Sentinel-1 Mission

The Sentinel-1 mission is the European Radar Observatory for the joint Copernicus initiative and aims to provide a constellation of two polar-orbiting satellites[2], which will operate day and night performing C-band Synthetic Aperture Radar (SAR) enabling imaging of the Earth's surface regardless of the weather. The two organizations responsible for this initiative are the European Commission (EC) and the European Space Agency (ESA). The mission has several objectives including a variety of different applications, such as land monitoring of forests, water, soil, and agriculture [50, 24, 36]. The polar orbit of the satellites results in higher revisit frequencies further away from the equator, e.g Sweden compared to Kenya.

2.2.1 C-Band SAR With Dual Polarization

The satellites use a single C-band SAR instrument operating with a center frequency of 5.405 GHz and dual polarization [50]. C-band radar is radar using the frequency ranges 4.0 to 8.0 GHz and is the portion of the electromagnetic spectrum allocated for commercial telecommunications via satellites, designated by the Institute of Electrical and Electronics Engineers (IEEE) [45]. SAR is an imaging radar commonly found on moving platforms, such as air crafts or satellites, which

[1] A great video showing how combine harvesters work can be seen here: Smarter Every Day: Farmers are Geniuses

[2] Unfortunately, Sentinel-1B has been out of service since December 2021 due to a power supply malfunction and therefore the data may be limited after the disruption [42].

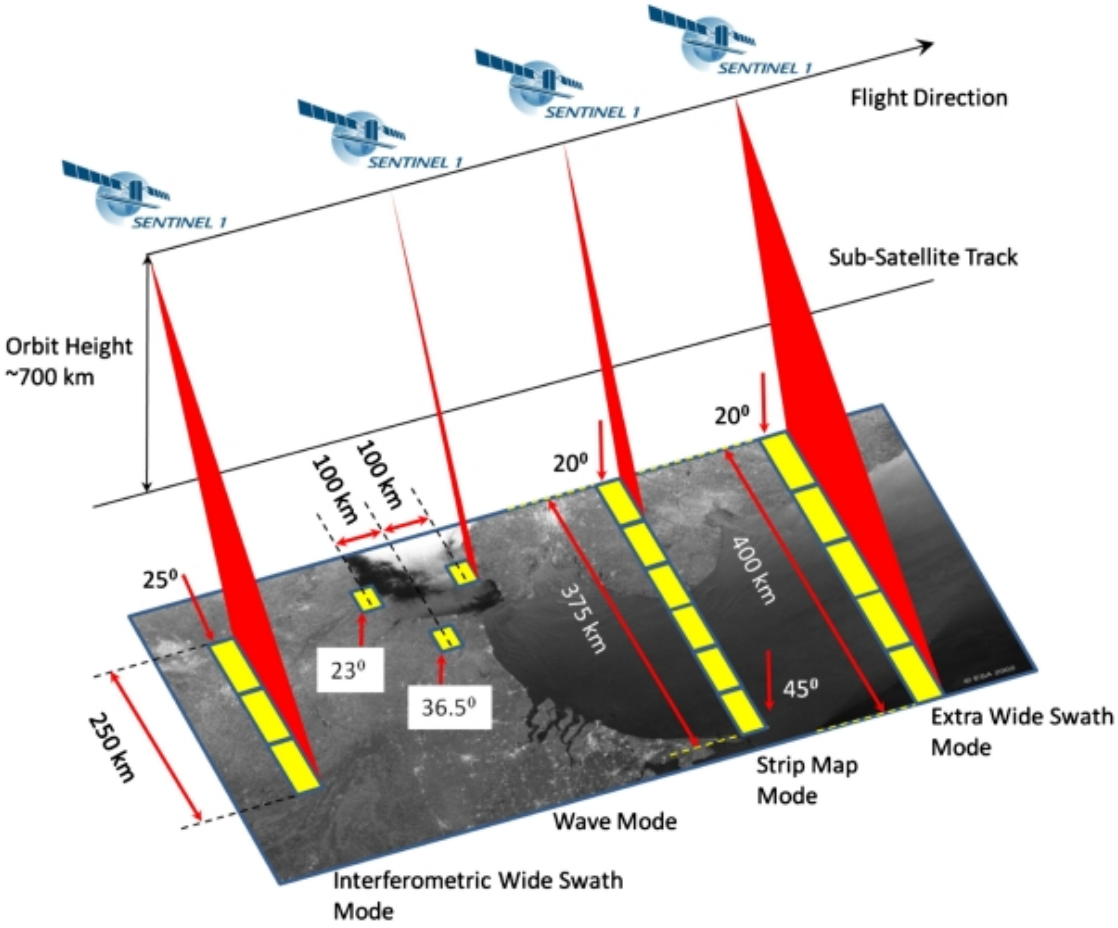

Figure 2.1: Sentinel-1 acquisition modes. As one can see there are different types of swaths that may be used for different applications. Retrieved from [50]

is able to produce high-resolution imagery with relatively small physical antennas. SAR works by transmitting a series of microwave signals and recording the backscatter from the Earth's surface [29]. Normally, to achieve high-resolution imagery a large aperture antenna is required for stationary radars. However, with a small and moving antenna successive recorded radar echoes may be processed and combined, giving the effect of a larger *synthetic* antenna aperture (hence the name) [45]. An advantage of SAR radar compared to optical imagery is the ability to operate regardless of weather conditions. Satellites using optical imagery are greatly impaired by clouds covering the target area [1]. The two polarization of the radars are horizontal (H) or vertical (V), and relates to the polarization of the transmitted or received signal. Different targets on the Earth's surface have distinct polarization signatures. The signatures reflect different polarization with different intensities of the target. For example, forest canopy backscatters have different polarization properties than sea surface backscatters [50].

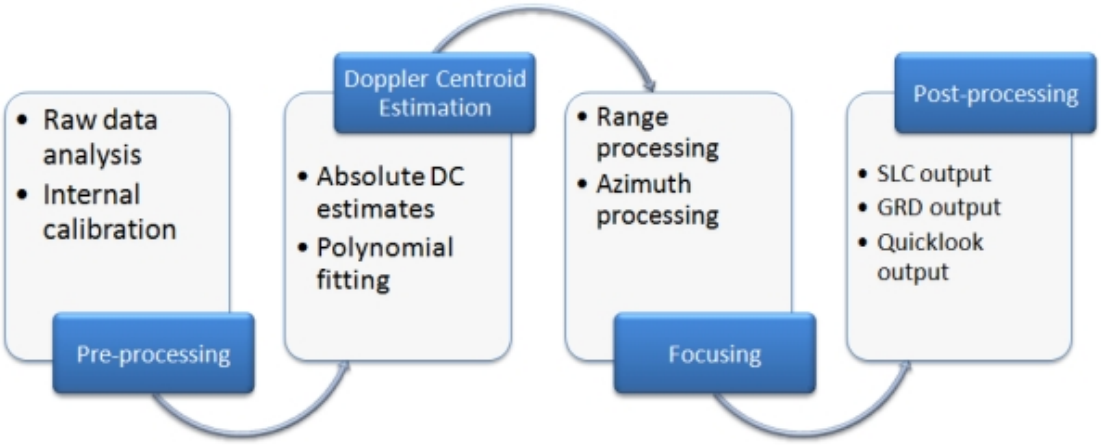

Figure 2.2: Level-1 processing chart of SAR data. Level-1 processing level is the main level for end users. Retrieved from [50]

2.2.2 Acquisition Modes & Product Levels

Sentinel-1 satellites have four acquisition modes with Interferometric Wide (IW) swath being the main acquisition mode over land (Fig. 2.1) and three different product levels with Level-1 being the most commonly used for end users [50]. IW mode supports VH and VV polarization with the highest resolution of 10.0 m [42]. However, the lowest resolution we managed to download from Sentinel Hub was 11.0 m. According to ESA, IW swath is the recommended main operational mode because it avoids conflicts, preserves revisit performance, decreases operational costs, and builds up a consistent long-term archive of data [50]. The three product levels are Level-0, Level-1, and Level-2, and relate to how much processing has been applied to the raw backscatter. Level-0 product contains raw, compressed, and unfocused SAR data, and is the basis for the other levels. Level-1 is the main data intended for users and is the level we used. It contains several processing steps, as seen in Fig. 2.2. We used the Ground Range Detected (GRD) product, which contains the detected amplitude from the SAR data and uses multi-looked to reduce the impact of speckling. The SAR backscatter can be visualized as an image, see Fig. 2.3. At the Level-1 product stage with orthorectification and radiometric calibration, the images are noisy and need further processing. Level-2 product is based on Level-1 product and consists mainly of geolocated geophysical products, such as Ocean Wind field [50].

2.2.3 Orthorectification & Radiometric Calibration

Additional processing of the Level-1 product such as orthorectification and radiometric calibration is needed for the SAR data to be able to measure true distance in the imagery and to convert digital numbers into physical units [47]. The process

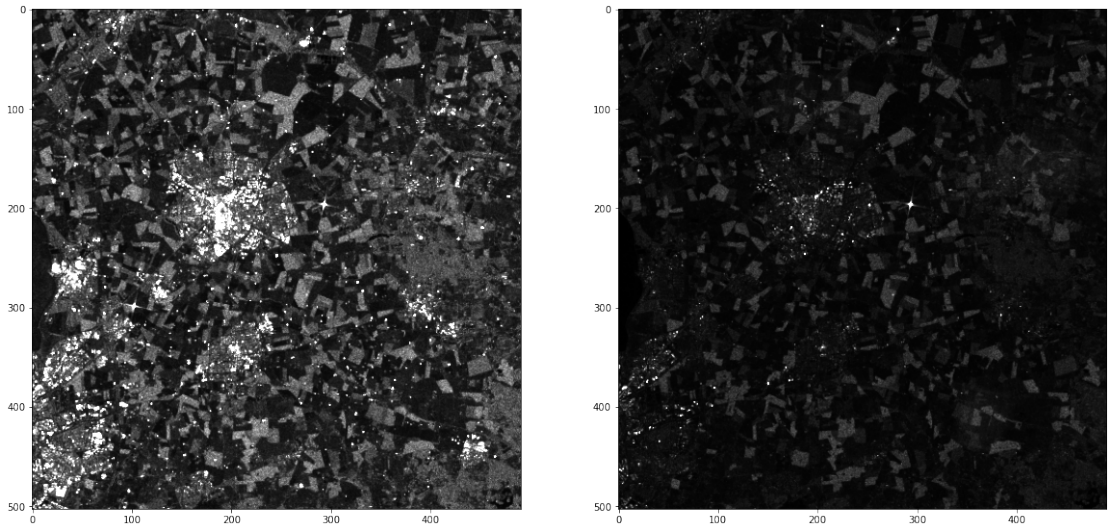

Figure 2.3: Level-1 SAR backscatter, VH (left), and VV polarization (right), with orthorectification and radiometric calibration. The bands have different intensity distributions which is why VH appears much brighter than VV.

of orthorectified satellite imagery is to correct the imagery for topographic relief, camera tilt, and lens distortion. Furthermore, a Digital Elevation Model is used as a 3D representation of the terrain's surface [42]. The result is imagery that can measure true distances.

2.2.4 Radio Frequency Interference

Sentinel-1 operates in the C-band where radio waves from other applications, e.g. missile guidance, can interfere with the originally transmitted signal from the satellite, causing radio frequency interference (RFI) [29]. An example of how RFI typically looks in the resulting backscatter can be seen in Fig. 2.4. Separating the part of the backscatter with RFI can be difficult, due to the values not being NaN or infinity. There are several proposed algorithms for detecting RFI for Sentinel-1 data [29]. Although the algorithms are publicly available, implementing them from scratch is not trivial. There is an alternative way of downloading Sentinel-1 data, using the Sentinel-1 Toolbox [50], where the data will be annotated with a flag if it contains RFI. The toolbox is significantly more difficult to use compared to the `eo-learn` python package [9], which we used to download the Sentinel-1 data.

2.3 Machine Learning

Machine Learning (ML) can be seen as a form of applied statistics with an increased focus on the use of computers (machines) to statistically estimate (learning) complicated functions and a decreased focus on providing confidence intervals for these

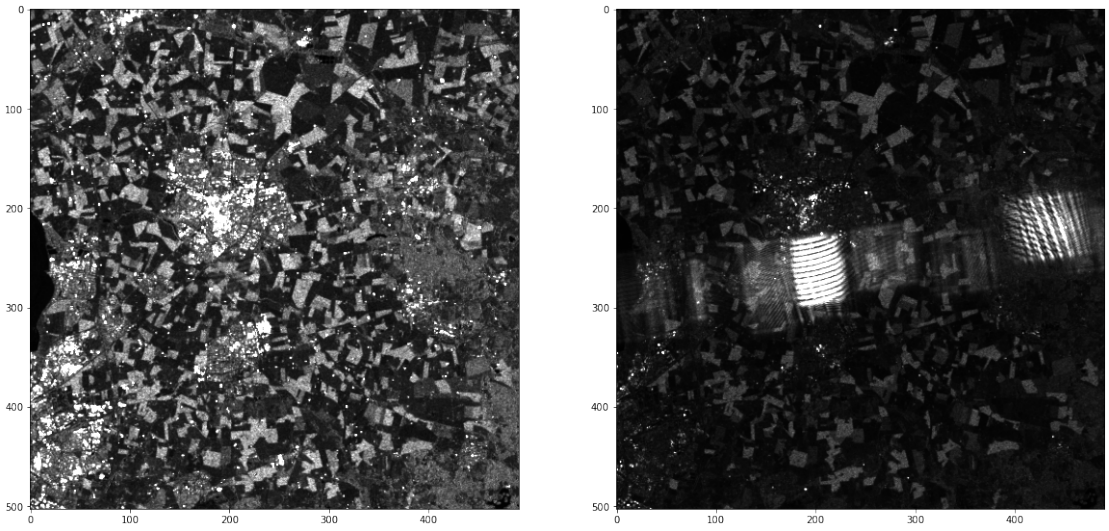

Figure 2.4: An example of Sentinel-1 data being affected with RFI, see VV band (right)

functions [13]. ML is a field of study within AI, and in general, when speaking about ML one tends to mean specific ML models or algorithms. A model is a description of a system using mathematical concepts and language. A learning algorithm is a process able to learn parameters of the model from data [13]. The learning process is often referred to as training of the model, and we will use the terms interchangeably.

A formal concise definition of learning is given by Mitchell [34]:

> 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.

A classical example, using the definition above, is the task of classifying handwritten digits. The performance measure can be the percentage of correct answers. The model (computer program) can be trained with experience from the MNIST data set [6]. The data set is divided into two parts: the handwritten digits stored as digital images (input data) and a label annotating what number the input is (ground truth).

In our thesis, the task is harvest prediction on winter wheat from fields in Skåne. The performance measures will be the root mean squared error, accuracy, and F1-score. There will be two main algorithms trained with supervised data (experience): Sentinel-1 radar backscatter, a multitude of weather-related data, topology classification of the terrain, and elevation data (input data); measurements of coordinate-wise average harvest in tonnes per hectare from the fields (ground truth).

2.4 Free & Open Source Software

This thesis would not be possible without free and open-source software, which we have used extensively. The main projects we would like to highlight and thank are:

- Python [16]
- Numpy [52]
- Pandas [53]
- Matplotlib [20]
- Tensorflow [15]
- Keras [11]
- Project Jupyter [10]
- LightGBM [31]
- QGIS [12]
- Open OnDemand [17]

2.5 Alvis

We were granted access to the Alvis cluster at Chalmers University of Technology [37] for this project. The cluster is used for ML and AI research, and consists of several compute nodes with powerful and expensive graphical processing unit (GPU) accelerator cards. We had many computationally heavy tasks, such as self-supervised despeckling and model training, which would not be viable to perform at this scale on our own machines due to memory and computational power limitations.

2.6 Related Work & Scientific Contribution

In this part, we will present work related to our thesis and how we contribute to science by adding to the authors' work. We will start by presenting a larger study in the Netherlands where they used Sentinel-1 to monitor several crops. Then, we will present a study done in Skåne where they used Sentinel-2 optical imagery to predict the harvest of winter wheat. Finally, a study from Lebanon will be presented, which used Sentinel-1 to monitor winter wheat.

2.6.1 Crop Monitoring in the Netherlands Using Sentinel-1

In a case study from the Netherlands, Sentinel-1 data was used to monitor the characteristics of sugar beet, potato, maize, wheat, and English ryegrass [24]. The monitored characteristics of the crops were the phenological stage, height, and key dates of interest. Comparing the time series of the radar backscatter with hydrometeorological data and field measurements, it was shown that Sentinel-1 data reflect moisture and structural changes associated with the phenological development of the crops during the growing season.

This study shows the tremendous potential of using Sentinel-1 data for crop monitoring and is a fundamental part of the science which this thesis builds upon. There are several similarities between this study and our thesis. We will also use Sentinel-1 data to get information about crops. The key difference is that we aim to predict crop characteristics (harvest can be viewed as a characteristic), instead of monitoring. This fundamentally changes the purpose of the Sentinel-1 data. In their study, they show that there exists a signal in the radar backscatter, by comparing the data with actual ground measurements and hydrometeorological data. Our goal is to leverage the signal in the Sentinel-1 data and combine it with other data to train ML models to predict the harvest.

2.6.2 Winter Wheat Prediction in Southern Sweden Using Sentinel-2

In a study from Skåne, Sweden, Sentinel-2 satellite imagery combined with local weather data, national soil databases, and local field data was used to train a Light Gradient Boosting Machine able to predict winter wheat yield with an accuracy[3] of 82 % [1]. The study presents several challenges of working with satellite data combined with other data where spatial and temporal scales differ. For example, combining satellite data which is continuous in space with soil measurements which are heterogeneous in space. Furthermore, the authors present one downside of using Sentinel-2 optical imagery - its dependence on cloud-free conditions to not block the imagery.

The study shows that harvest prediction on winter wheat in the Southern area of Sweden is possible. A benefit of this thesis is that we will get data from the same source as the authors of the study and from the same area of Sweden. We will build upon this research by using Sentinel-1 data, which does not suffer from

[3]Other metrics for regression were also reported by the authors. However, these are from training and not testing of the models, and as such, not comparable with the errors we present as they come exclusively from testing.

cloudy conditions. Another addition to the study is the use of the topological classification of the fields. Finally, we will also test a Feedforward Neural Network model and compare it with the model they used.

2.6.3 Winter Wheat Phenology Monitoring Using Sentinel-1

The ability to use Sentinel-1 data to detect important winter wheat phenological phases and harvest was thoroughly investigated in a study from Lebanon [36]. Temporal variations of the Sentinel-1 backscatter were analyzed as a function of phenological phases' (germination, heading, and soft dough) and harvesting dates, and it was shown that the variations of the Sentinel-1 data could be used to estimate the dates of these important phases. The authors thoroughly investigated how the different polarization (VH, VV) and the ratio VV/VH with different incidence angles could be used to predict the important dates. It was shown that different signals of the backscatter were more efficient for certain tasks, e.g. the ratio VV/VH was shown to be the best predictor of the germination and harvesting phase.

The authors speak at length about the importance of these dates for the farmers, as phenological phases may require interventions by farmers and decision-makers. Interventions include irrigation, fertilization, pesticide application, and yield handling. Mapping the dates correctly in a real-time setting can lead to farmers being able to perform the interventions at the correct time. For example, applying irrigation past the appropriate time window may lead to increased fungal diseases. The dates are also important from a crop modeling perspective. For example, the germination phase is a critical phase for the farmers, as this date is the starting point for the growing degree days accumulation in crop simulation models (used to estimate the season's production outputs).

The study is of great value to us, as its main focus is the same crop we aim to analyze. Furthermore, the usage of IW swath Sentinel-1 GRD data is the same as we will use. Similarly to the study from the Netherlands [24], a difference between our thesis and their study is that our aim is not to monitor the crop but to predict harvest using ML models trained with Sentinel-1 data combined with other data. Another key difference is that our study is done in Sweden, which has a different climate compared to Lebanon.

Chapter 3

Methodology

This chapter describes how all data was obtained, how it was preprocessed, and how it was used to construct data sets for the machine learning models. Furthermore, we also describe our models, metrics, and features in the final data sets. We collected the most important parts of our methods in a GitHub repository [49].

3.1 Sentinel-1 Data

The Sentinel-1 data was captured with acquisition mode IW swath, Level-1 product, GRD, resolution 11×11 m^2, with added orthorectification, and radiometric calibration over fields in Skåne (Fig. 3.2) for the years of interest and main time period. The years of interest were 2017, 2018, 2019, and 2020 and the main time period was April 1st until July 31st (see Fig. 3.1). We selected the years because of the availability of data from the fields (i.e. measured harvest). We selected the main time period due to previous work on winter wheat harvest prediction from Skåne [1], where they show that the main time period contains the growing dynamics of winter wheat, see Fig. 3.3. For the main time period, we got 2-3 samples of Sentinel-1 data per week. To build our data sets, the first sample with defined values for the VH and VV band, i.e. the intensities not being NaN, was picked for a given week.

Our downloaded Sentinel-1 data was a time series of VH and VV backscatter for each year over the main time period, with around 2-3 days between each backscatter. The backscatter can be viewed as images, where each pixel value relates to the 11×11 m^2 backscatter intensity for a part of the Earth's surface, see Fig. 2.3.

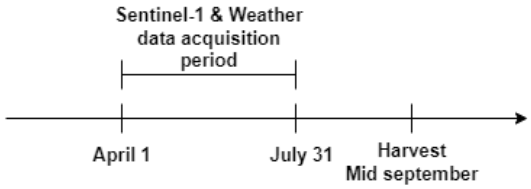

Figure 3.1: The Sentinel-1 and weather data was collected from the first of April until last July. The end date is about a month before the harvest which occurs in mid-September. The data collected had a higher resolution in time than one week, and as such, it was resampled into weekly samples. For Sentinel-1, this was done by selecting one day for the whole week, and for weather, the daily measurements were either averaged or accumulated.

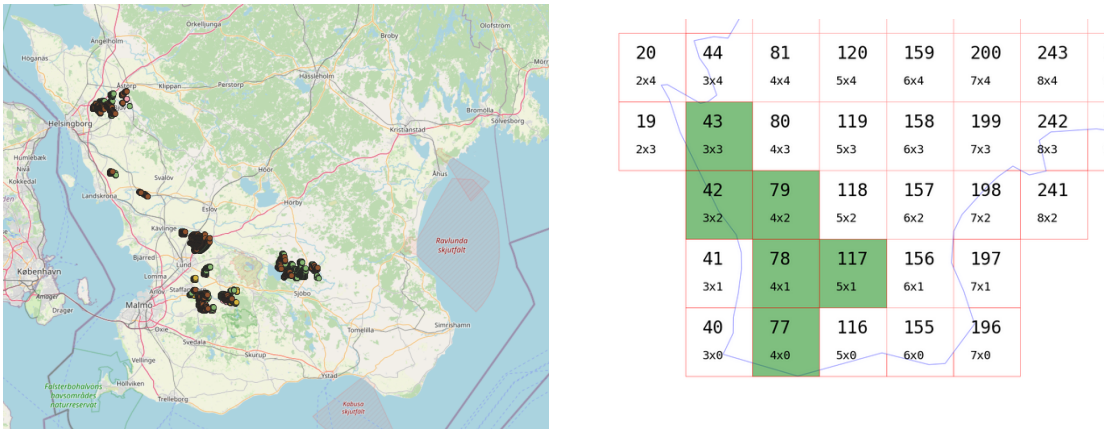

Figure 3.2: A geographical overview of the data. The left figure shows the field points and the figure to the right shows the patches of Sentinel-1 data. The patches used are highlighted in green and covered all the field data. As the field data was in a coordinate grid, extra steps were needed to get the corresponding VH and VV backscatter from the larger patch, which also contained backscatter from outside of the fields.

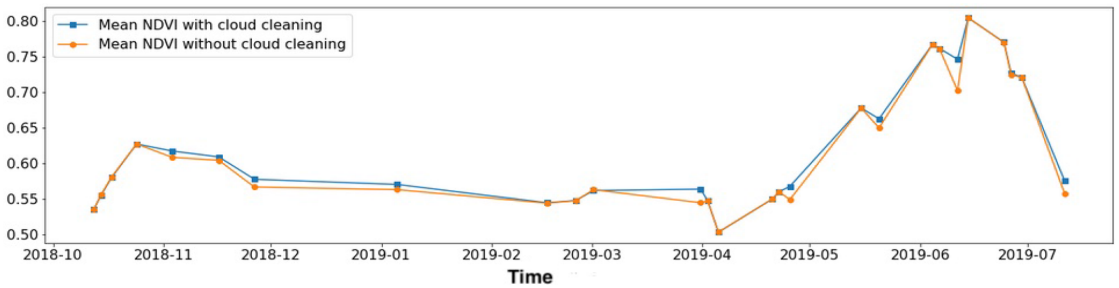

Figure 3.3: The normalized difference vegetation index from [1], showing the increase in the vegetation of winter wheat starting in April until the end of July.

3.1.1 Derivatives of VH & VV

To help the ML models perform better, feature engineering was performed to create derivative features based on the VH and VV data. These derivatives, referred to by us as *indices*, were computed by simple mathematical operations. The simplest was the *ratios*, which simply was VH/VV and vice versa. The other indices were slightly more complex:

$$\text{RVI} = \frac{VH}{VH + VV} \tag{3.1}$$

$$\text{M-RVI} = \frac{VV}{VH + VV} \tag{3.2}$$

$$\text{MI0} = \frac{VH - VV}{VH + VV} \tag{3.3}$$

$$\text{MI1} = \frac{VH - VV}{VH} \tag{3.4}$$

It should be noted that there were no divide-by-zero issues with any of the indices, as after despeckling with SAR2SAR (see Section 3.1.2) all VH and VV values were larger than 0 or NaN.

3.1.2 Despeckling

The benefit of SAR satellites' ability to penetrate clouds and operate day and night does not come without drawbacks. SAR data is frequently corrupted by speckle noise which must be dealt with before it is interpreted by any human or model. Speckle noise is caused by a physical limitation of SAR satellites, occurring when the radar is tasked with summing different elementary incoherent scattered radio waves from within a single-resolution cell.

What makes speckle noise extra tricky to work with when compared to most other noise is the fact that it is multiplicative noise, meaning that the speckle noise u is modeled as multiplication with the underlying reflectivity v resulting in a final intensity $w = vu$. This makes simpler noise reduction methods unusable since they usually model additive noise.

There exist traditional methods for speckle filtering, such as Gaussian filters, median filters, or Lee filters. The problem with these filters is that they distort the data such that information is lost. Frequently, filtering these images leads to excessive blurriness or introduces other unwanted distortions and artifacts.

Recently, an ML approach for despeckling SAR data called SAR2SAR was developed [3]. The method is an extension of an earlier but also recently proposed idea called Noise2Noise [28]. These methods enable the restoration of noisy data

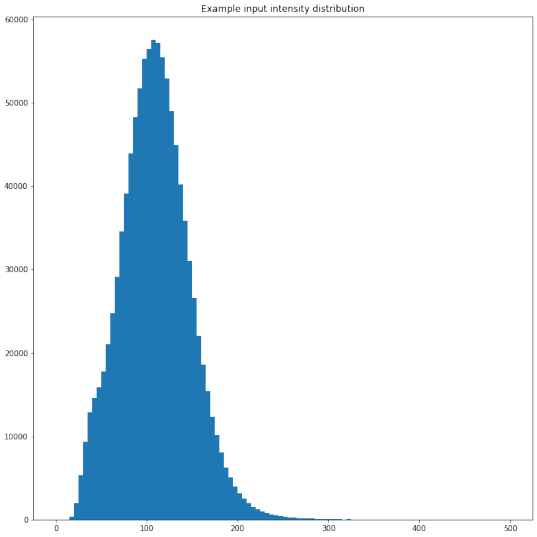

Figure 3.4: Intensity input distribution for one of the provided example images which came with the SAR2SAR repository

without ever having access to any clean examples. This is possible through the use of self-supervised learning. In rough terms, SAR2SAR learns to despeckle an image x by distorting it with additional synthetic speckle noise to create two new images y_1, y_2 such that $y_1 = f_1(x)$, $y_2 = f_2(x)$. The model then uses these new images to train a U-Net despeckling network using y_1 as input and y_2 as the label (ground truth). Since they are the same image with different amounts of noise, the network learns to remove speckle noise.

Given that we have access to many noisy examples of Sentinel-1 SAR data, but no clean data, this method suited our needs perfectly. Since we lacked the time to train a SAR2SAR model using data Sentinel-1 data from Skåne, we decided to use the pre-trained model provided by the SAR2SAR team [8]. Before we could do this we had to adapt our data to fit the model. Luckily for us, the SAR2SAR team provided examples of input images to the model. We decided to look at the intensity distribution of the examples (see Fig. 3.4) and construct a transformation that would map our data to have a similar intensity distribution. The transform we came up with to transform each pixel x in a backscatter X from Sentinel-1 was

$$T(x,p) = \begin{cases} 20\log_b(50 \cdot 2x + 2) & \text{if } p = VH \\ 20\log_b(50 \cdot 10x + 2) & \text{if } p = VV \end{cases} \qquad \forall x \in X\ ,\ b \in [1.5, 2]$$

The "magic numbers" in this expression were there for a specific purpose. Firstly, we did not want to compute the logarithm of zero, and we also did not want pixels to be less than or equal to 0 after the transform. This would have made later processing more difficult and tedious, so we added a constant of 2 to the

expression to avoid these situations. Secondly, we needed to scale the VH and VV bands separately because they had different ranges, excluding outliers. These ranges were approximately VH $\in [0, 0.5]$ and VV $\in [0, 0.1]$, so we scaled VH by 2 and VV by 10 to give them both an approximate range of $[0, 1]$.

The scaling numbers 50, 20, and the log base $b = 1.8$ were found by experimentation where the goal was to construct an intensity distribution similar to the intensity distribution of the provided example images. We noted that the whole interval $b \in [1.5, 2]$ yielded good despeckling. We confirmed the transformation by assuring that we had a good despeckling effect in the output and by plotting the distributions for some examples. Furthermore, we also tested and compared the performance of our ML models with and without despeckling and regularly saw an increase of about 5-15 % in accuracy and F1-score when using the despeckled data.

See Fig. 3.5 which shows how this transformation affected the intensity distribution of the VH and VV bands. The results from despeckling with SAR2SAR were significant, see Fig. 3.6 for examples of this.

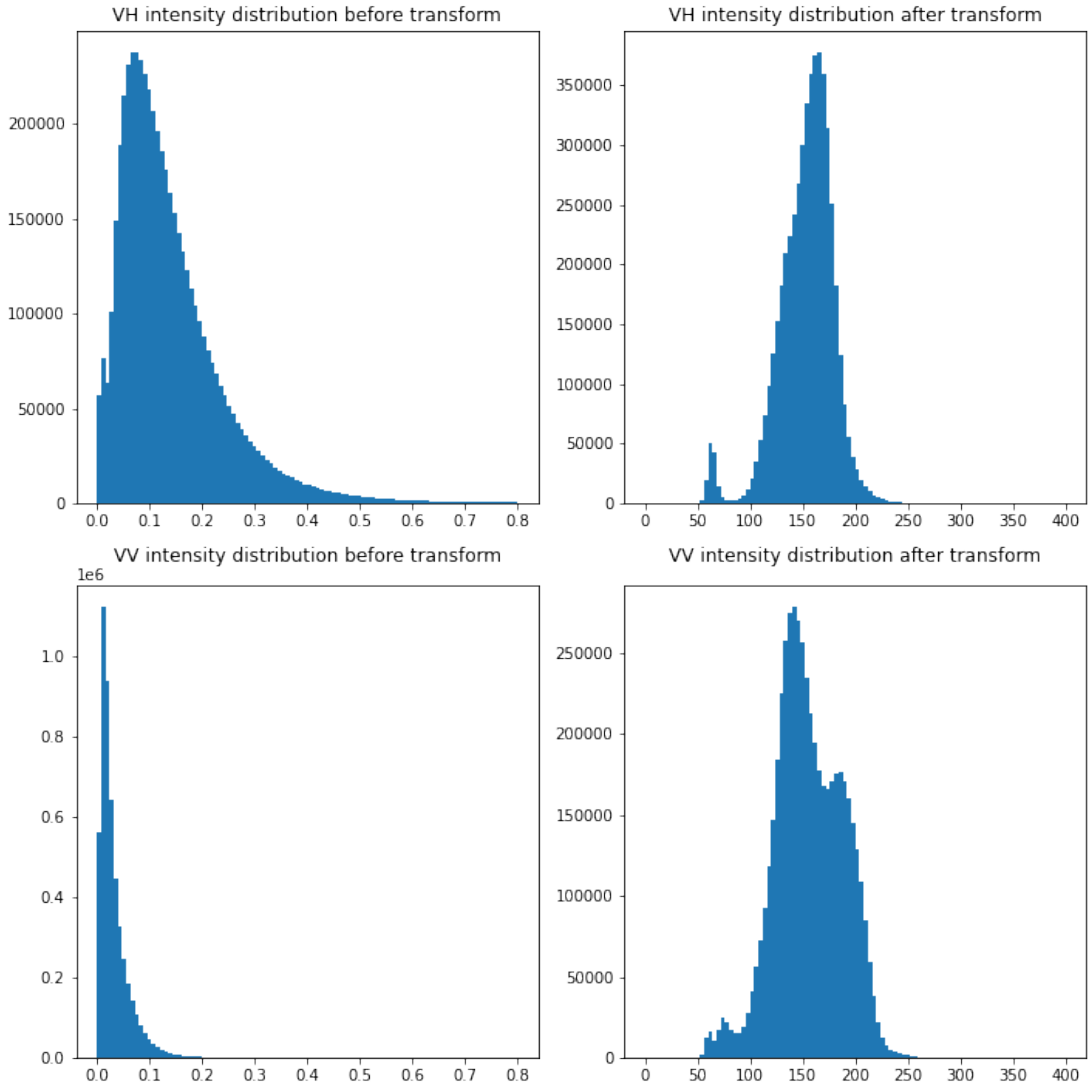

Figure 3.5: The effect of the transformation $T(x, p)$ on the intensity distributions. The left column is the distribution of the raw data while the right column is the distribution of the transformed data. Note that the scales shift significantly. Also, note that the mean is slightly higher than the example distribution (Fig. 3.4). However, since we had good despeckling results when using these images we decided that we could ignore this.

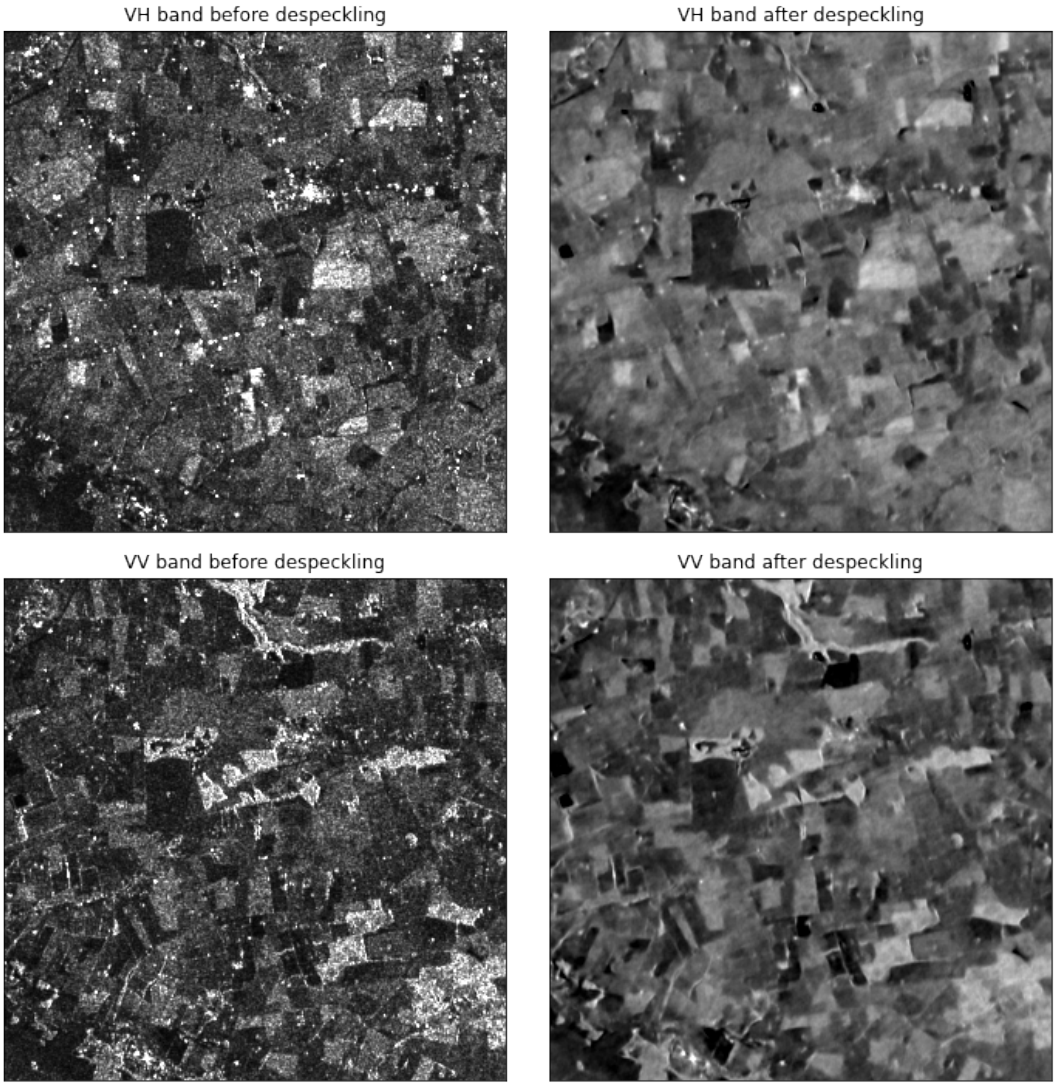

Figure 3.6: The results of despeckling with SAR2SAR for both bands. Some sharpness is lost in the process, but compared to traditional methods, this is a huge improvement.

3.1.3 Filtering RFI

As an optional extra step of preprocessing Sentinel-1 data, we wanted to see if we could remove some of the RFI which can be seen in Fig. 2.4. The RFI varies a lot in intensity, density, and size as can be seen in Fig. 3.7.

Ideally, we would probably like to both detect and filter the RFI with an ML model on its own, but this solution is outside of the scope of this thesis and we decided to use a much simpler method based on morphology. The core idea of our method is that the RFI vanishes when an image is averaged over several time steps. We detect the RFI X_t^{RFI} by computing the difference between a current image X_t and its average over time X_{mean} such that

$$X_t^{\text{RFI}} = |X_t - X_{\text{mean}}| = \left| X_t - \sum_{t_0 \leq t \leq t_1} X_t \right|$$
$$\text{where} \qquad 0 \leq t_0 \leq t_1 \leq T, \qquad I = t_1 - t_0.$$

where T is the final time step and I a sampling interval size. We have some edge cases when choosing t_0 and t_1 at the start and the end of the time series. We deal with these cases by averaging more forward or more back in time as needed from the current image X_t to maintain the interval length I. The resulting image X_t^{RFI} contains most of the RFI in X_t if it had any. We can then remove it from X_t with the help of morphology. The pseudocode for the full algorithm can be seen in Alg. 1. The morphology filter sizes and thresholds found in the pseudocode were found by trial and error in every step of the process until sufficiently good results were achieved. Fig. 3.8 shows the effect of the RFI-filter on the data.

Algorithm 1 RFI Filter

$X \leftarrow$ Load_Images()
for $t \in [0, ..., T]$ **do**
  $X_t \leftarrow X(t)$
  $X_{\text{mean}} \leftarrow$ Compute_Mean(t, X)
  $X_{\text{RFI}} \leftarrow |X_t - X_{\text{mean}}|$
  $X_{\text{tresh}} \leftarrow X_{\text{RFI}} >$ Noise threshold
  $X_{\text{eroded}} \leftarrow$ Erode(X_{tresh})
  $X_{\text{label}} \leftarrow$ Label connected regions in X_{eroded}
  Outlier_labels $\leftarrow$ Area$(X_{\text{label}}) >$ Area threshold
  $X_{\text{Outlier}} \leftarrow$ Merge outliers into single binary image
  $X_{\text{Outlier}} =$ Dilate(X_{Outlier})
  $X_{\text{Final}} \leftarrow X(X_{\text{Outlier}}) =$ NaN

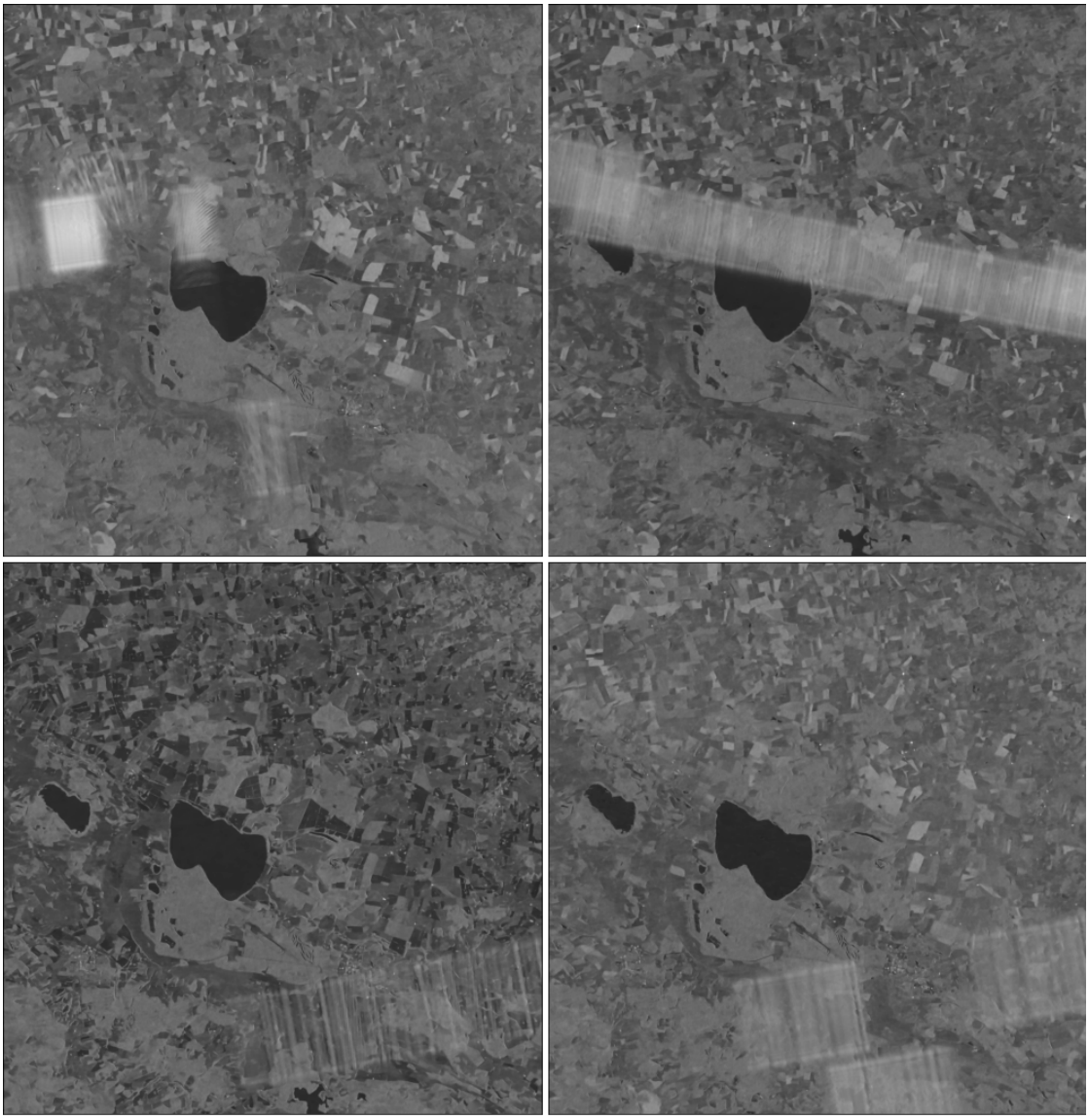

Figure 3.7: Examples of RFI noise in the VV band (which contains most of the noise). Notice how the RFI varies in intensity, density and size. Some images have large areas covered by low-density noise while others have small areas covered by high-density noise and so on. This makes it hard to find a single method that captures all variations of RFI.

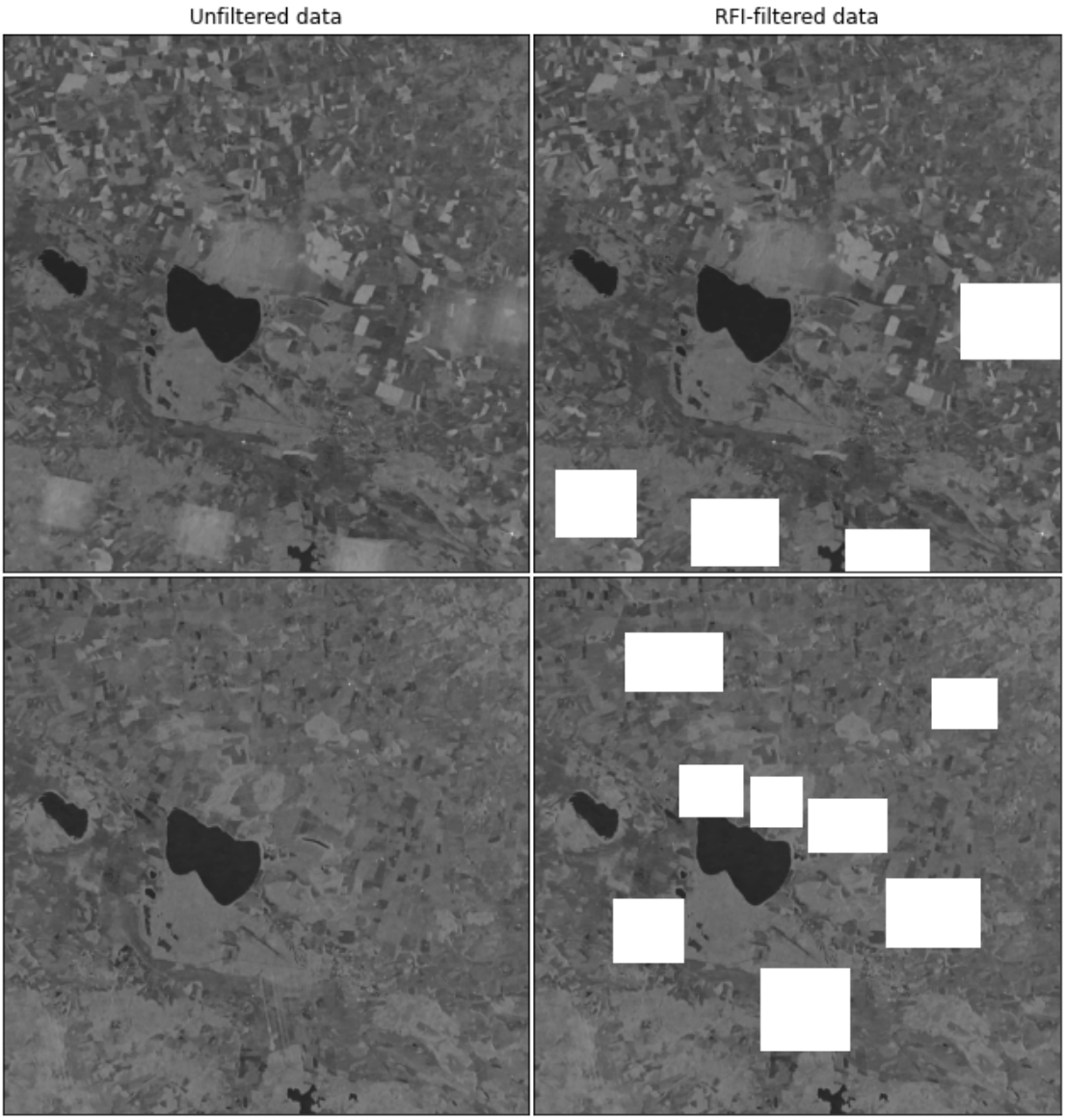

Figure 3.8: The effect of the RFI filter on the dataset. The top row depicts a good example. While one spot was not detected, it still removed the worst regions with quite good precision. The second row depicts the worst example we could find, where the filter removes a lot of data that has no visible RFI at all. Fortunately, we could only find a single case of this happening at this scale. Usually, the filter only removes RFI while sometimes missing a spot. It rarely removes good data.

3.1.4 Visualization

Time series over an area of interest can easily be visualized with plots, revealing important information about the data. From our field data, the coordinates of the fields (Fig. 3.12) were known, and therefore visualization of only VH and VV from the fields was possible. This was done by picking the VH and VV values for the coordinates and their 1-distance neighbors. VH and VV values outside the fields, e.g. of a city, could therefore be filtered away and were not part of the visualization, see Fig. 3.10 and Fig. 3.11. The figures show the result of the mean of all VH and VV on fields for a certain patch. The discontinuity of the curve in the plots comes from the fact that we have a lot of NaN values in our data. Fig. 3.9 depicts what the NaN values look like in the satellite data.

We refer to a time series of VH and VV data as a patch. Some patches do not have the problem with periodically reoccurring NaN values (see Fig. 3.10 while others do (see Fig. 3.11). We believe that the cause of these NaN values comes from the patch only partially being contained inside a common Sentinel-1 orbit operational range.

The *good* patches (78 and 117) do not have the problem of reoccurring NaN values. Fig. 3.10 shows a time series for fields within patch 117 where we can see a clear downward trend in the VH band from April until the middle of May followed by a clear upward trend until the end of the time period. For the VV band, there is a small upward trend for the whole time period. VH seems to have a higher variance and range of values in general.

The *Bad* patches (42, 43, and 79) contain more missing values, leading to a discontinuity in the curves, see Fig. 3.11. The cause of these gaps was that for a certain date all VH or VV values on the fields were NaN. The general trend for VH and VV seemed to be similar to the good patches.

Effect on ML Models

Ideally, all patches would look similar to the good patches, with low amounts of discontinuity. However, the bad patches do not necessarily have a large impact on our ML models. For each given coordinate of measured harvest, we created a feature vector containing the VH and VV data (also potential indices based on VH and VV) from one date in each week. Because only one VH and VV sample was needed for a week, we tended to be able to find one such date for all patches where the values of VH and VV existed. As such, the missing values in patches did not impair our ability to create large data sets and therefore we believe the effects were limited. Furthermore, the general trends seemed to align between the good and bad patches, which further supports our belief.

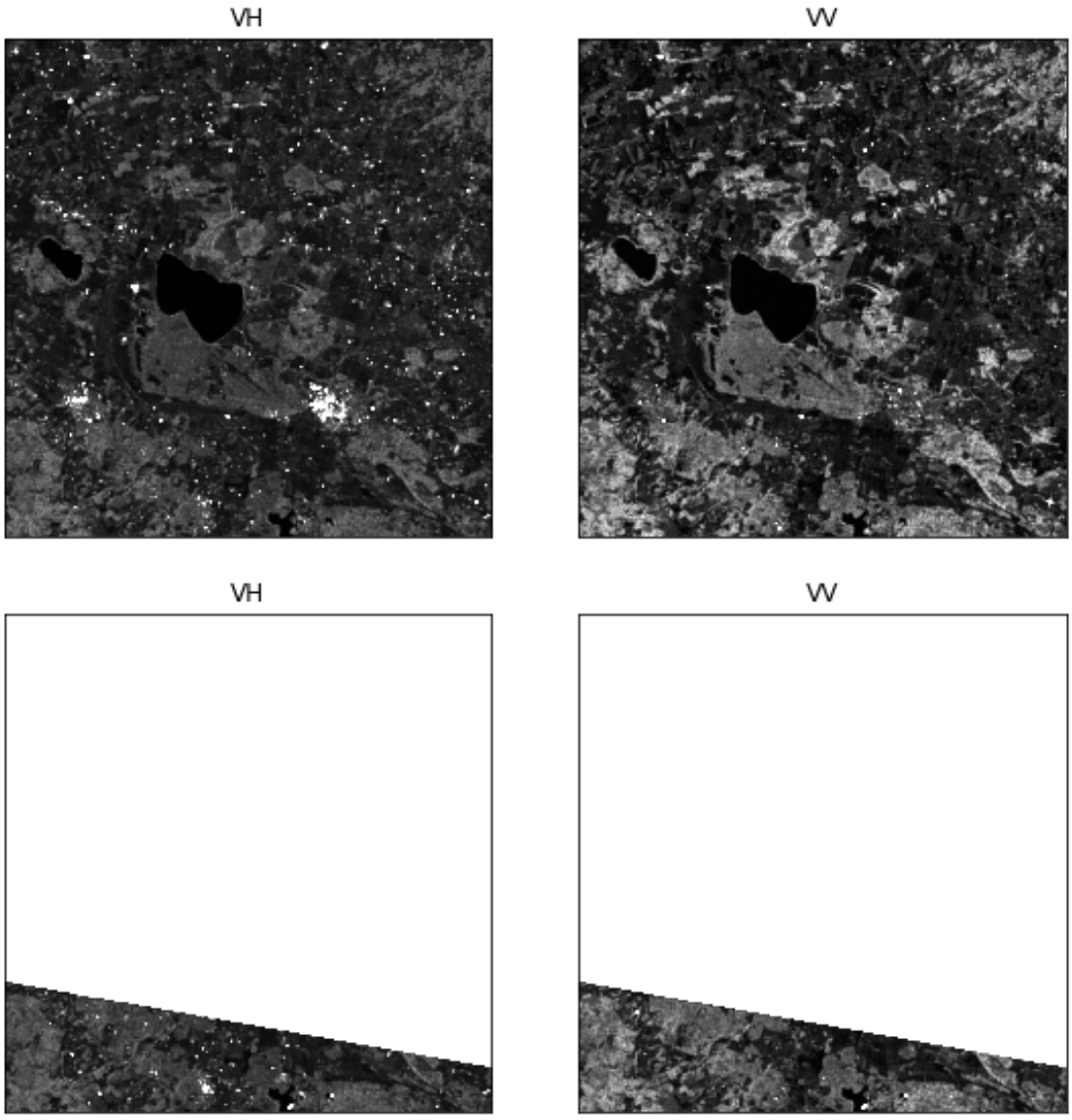

Figure 3.9: The data contains a lot of time steps which consist mostly of NaN values. Since the data which is not NaN was fine we suspect that this comes from the fact that the satellite can not always cover the full area we requested to download on every orbit, but Sentinel Hub still sends us whatever data was available.

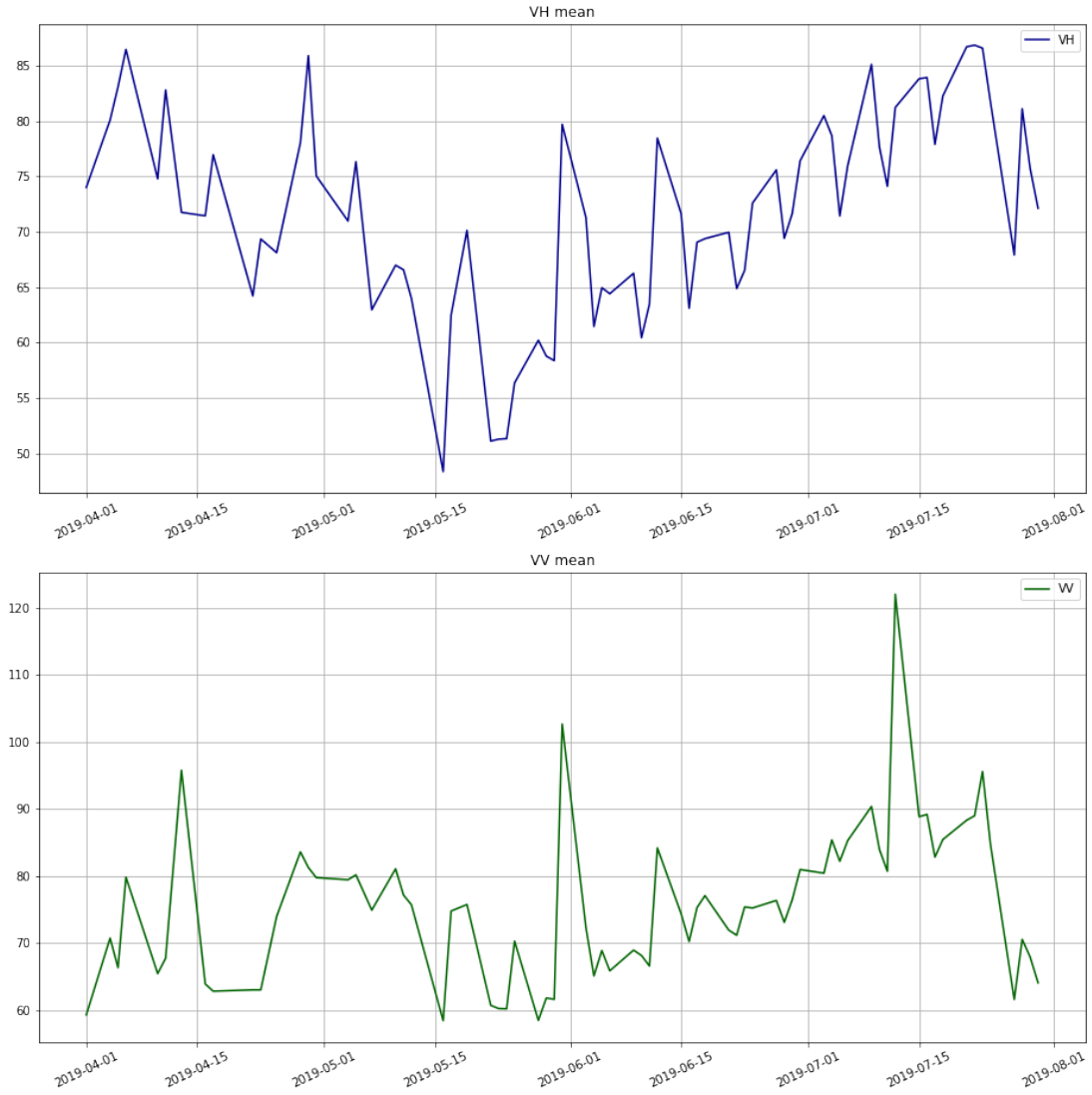

Figure 3.10: This figure shows how the mean of all VH and VV values on fields from patch 117 (see Fig. 3.2) vary from the start of April to the end of July 2019. For the VH band on this patch, there seems to be a downward trend from April to the middle of May, where it then turns into an upward trend until the later dates of July. There seems to be a slight upwards trend for VV with low variance and occasional spikes over the time period. In an ideal situation, all other patches should have similar continuous plots but this was only true for patch 78. For the other patches, this was not the case (see Fig. 3.11) and several patches had dates where fields VH and VV values were all NaN.

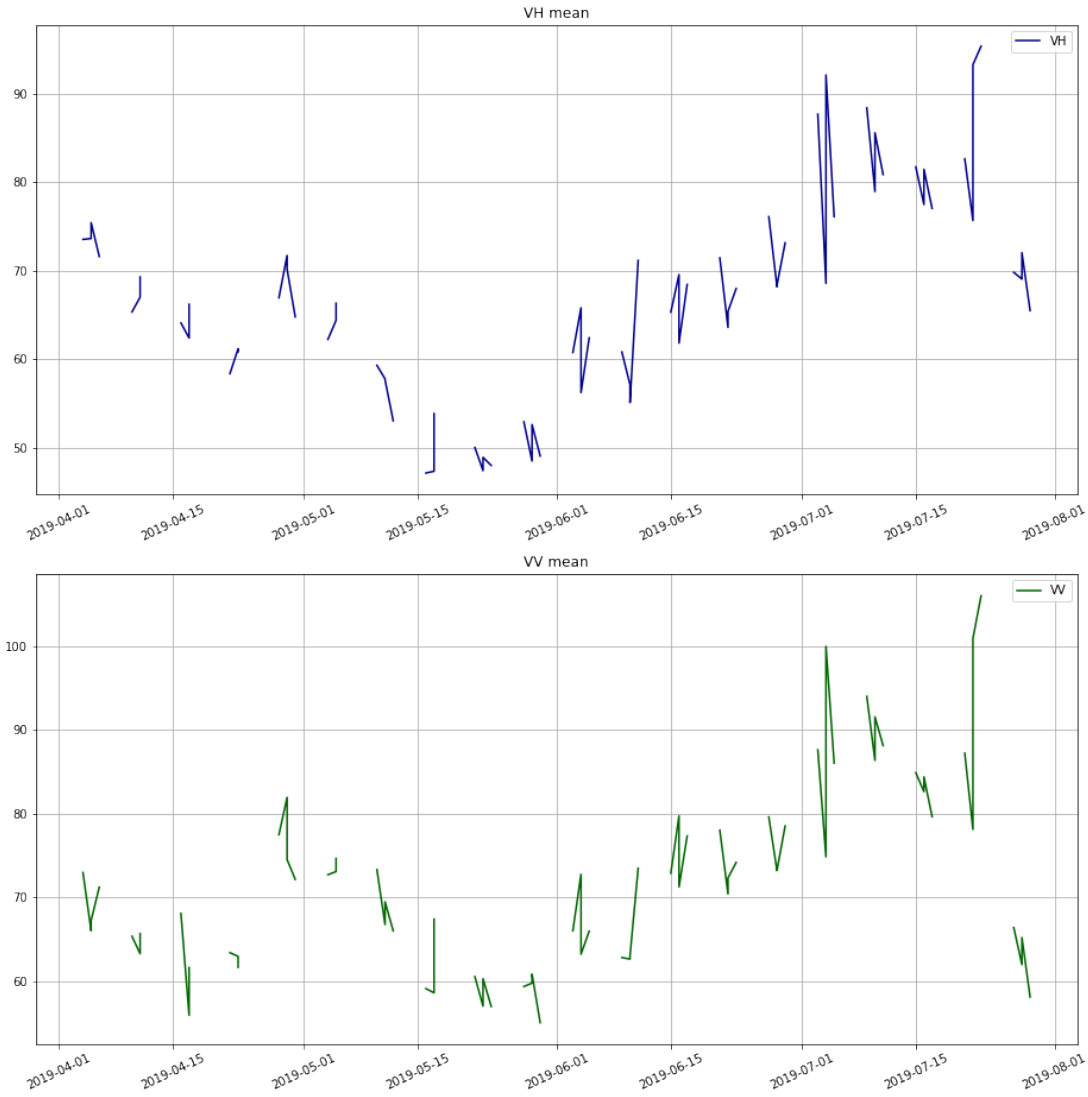

Figure 3.11: This figure shows how the mean of all VH and VV values on fields from patch 43 (see figure 3.2) vary from the start of April to the end of July 2019. As in figure 3.10, we see a downward and upward trend in the VH band and an upward trend for the VV band. The discontinuity in plots is caused by the patch having only NaN values over the fields. This problem was seen over all the years, and also occurred for patches 42 and 79.

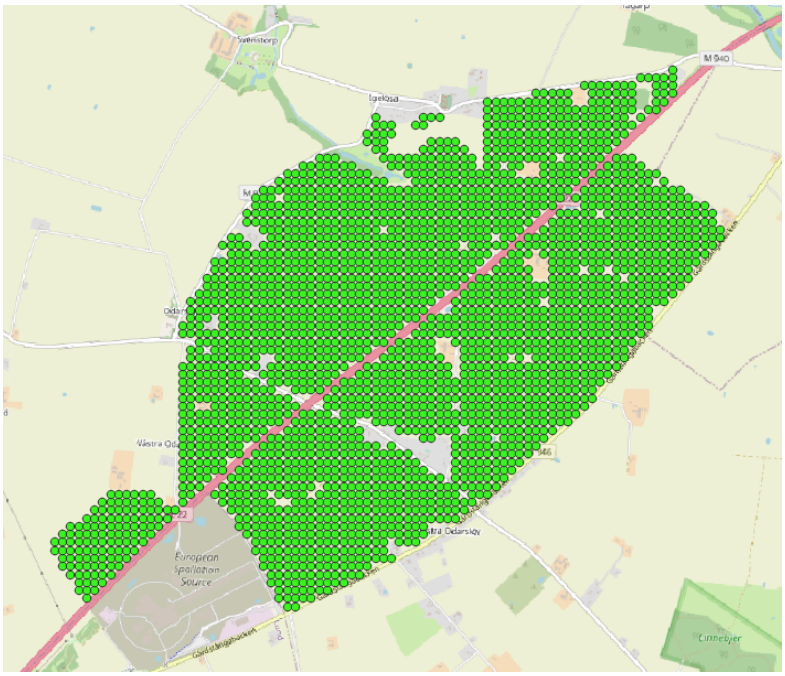

Figure 3.12: Example section of the harvest data grid when viewed in Qgis. Note that the points only cover fields and not any unnecessary regions

3.2 Harvest Data

The raw harvest data is gathered by combine harvesters[1] and can be thought of as discrete data points in space and time which measure the harvest in tonnes per hectare. Using the raw data points directly is not coveted, due to issues of uneven distribution, overlap, duplicated data points, and other issues. To deal with this, T-kartor has supplied us with two[2] coordinate-based grids containing the summarized harvest data for an area. Each grid summarizes the harvest within grid blocks with areas of 50×50 m^2 or 22×22 m^2 (see Fig. 3.12). The summarizing in each grid block is done by averaging the data points within a 25 m or 11 m radius of the center coordinate, resulting in the average data point harvest (tonnes per hectare) for this area (see Fig. 3.13). The grid coordinates establish the foundation for our data sets. All data coming from other sources are sampled based on these points. This makes the sampling process easy and by design, gives us a way to control data leakage.

The amount of harvest data varies over the year. Fig. 3.14 shows how much data we have for each given year and resolution. Notice the large increase in data when using the 22 m grid compared to the 50 m grid (more than 4x).

[1] A video showing how combine harvesters coordinate-wise approximate the yield can be seen here: Smarter Every Day: Farmers are Geniuses

[2] Late into the thesis, we also got a 12×12 m^2 grid structure and the results for this grid are informally shown in the Appendix.

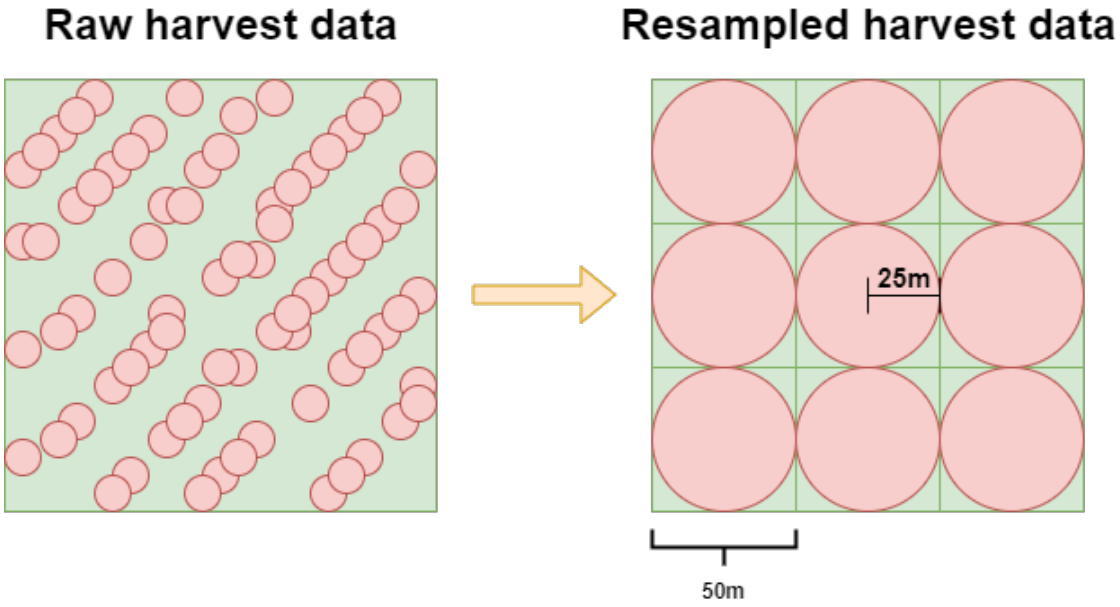

Figure 3.13: Illustration of the unevenly distributed points in the raw harvest data and how it was resampled into a 50×50 m^2 grid.

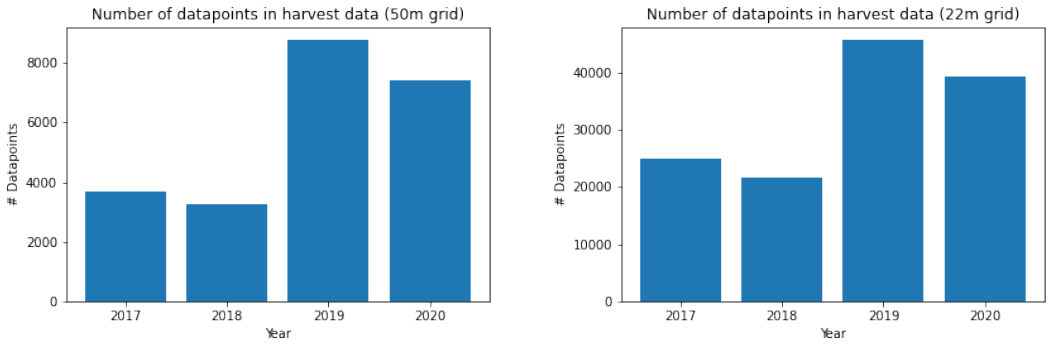

Figure 3.14: Data set size for each year with the 50 and 22 m grids.

3.3 Weather Data

We were granted access to an API for fetching weather data made by Niftitech. The API allows us to access a database with historical weather data collected from weather stations all over Sweden, Fig. 3.15 shows these weather stations in Skåne. The database is a collection of 4 public databases called Lantmet, Trafikverket, Fieldsense, and partly SMHI.

To use the API we have to provide it with a coordinate, a time period, weather metrics, and a maximum allowed sampling radius. The API then looks up all weather stations within the allowed search radius and sends us the averaged data collected at these stations divided into daily buckets, such that we get the mean or accumulated sum of the metric for each day within the given time period. The weather metrics we decided to use were.

- Temperature
- Solar radiation intensity
- Precipitation
- Wind speed
- Humidity percentage

Since the density of the weather stations is much smaller than our harvest data, we thought that it was unnecessary to fetch weather data for each individual point since nearby points will use data from the same weather station and thus have identical weather data. Hence, to simplify the process, we only fetched data for the mean coordinate of each field and then later assigned each point within a field the same weather data. Finally, since we do the machine learning on a weekly timescale we decided to resample the daily weather data into weekly weather data to fit with the rest of our data set (see Fig. 3.1).

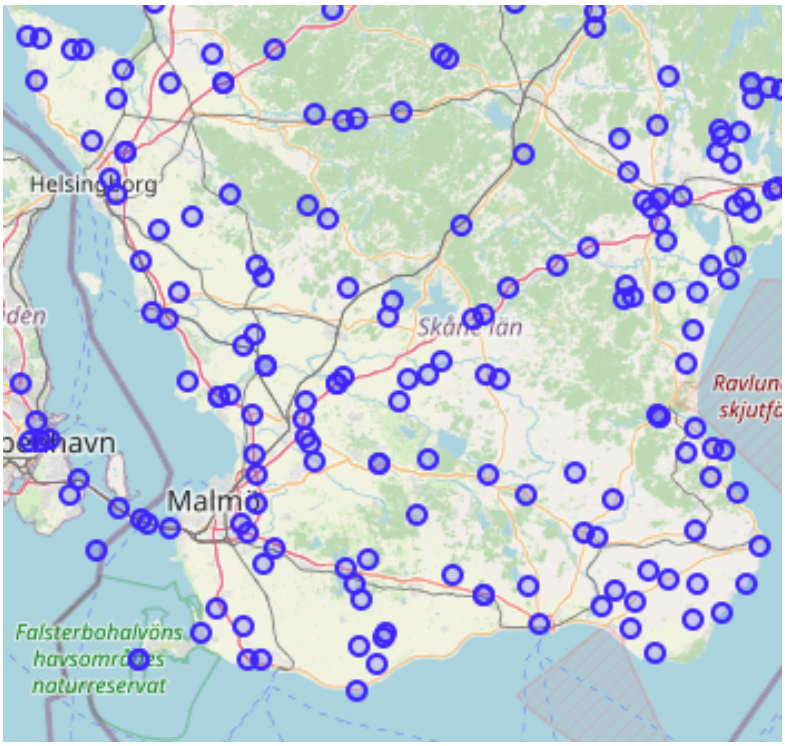

Figure 3.15: Positions of the weather stations in Skåne used by Niftitechs weather API, note that there are large regions that do not have a weather station in the database.

3.4 Elevation & Topological Classification

Skogsstyrelsen has gathered elevation data with 10 m resolution which covers a majority of Swedish landmass by using laser scanners attached to airplanes [44]. This data has been extracted for the fields of interest (see Figure 3.16) and then later processed by Grass GIS, more specifically the geomorphon terrain classification algorithm of Grass GIS [19]. This algorithm classifies each pixel of a pixelated terrain into one of ten integer classes based on each pixel's surroundings by considering the relative elevation difference between itself and its neighbors. More specifically, these classes are flat, peak, ridge, shoulder, spur, slope, hollow, footslope, valley, and pit,

Fig. 3.17 shows a visual representation of these classes and Fig. 3.18 shows the resulting terrain classification raster. To use the raster data in the model we vectorize it by sampling the raster at the coordinates where we have the harvest data. When creating the topology features, we represent them with one-hot-encoding instead of their integer value. We thought this made sense since classes with nearby integer representations are not necessarily topologically close to each other.

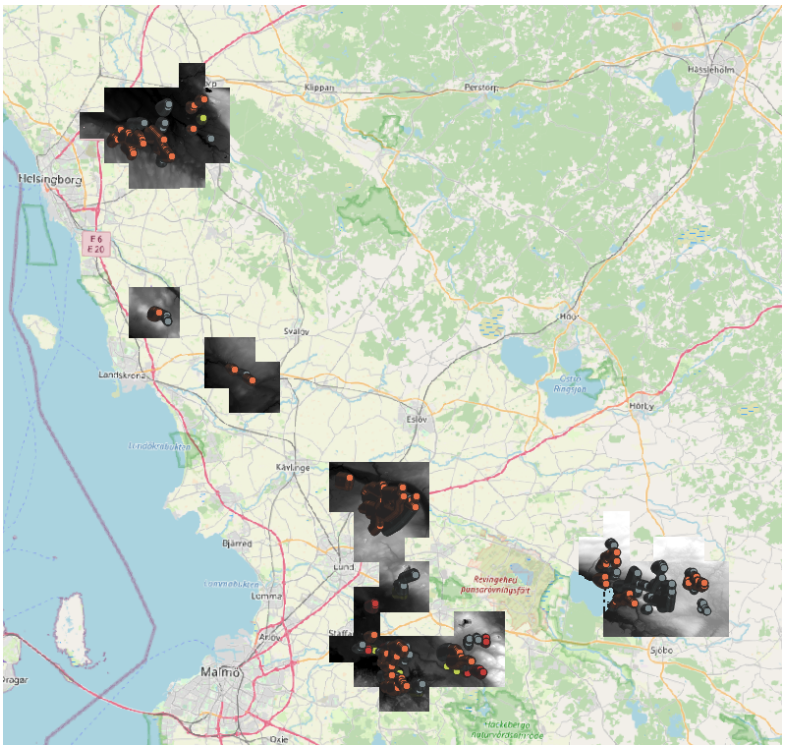

Figure 3.16: Selected elevation data from Skogsstyrelsen for our fields of interest with our harvest data points on top.

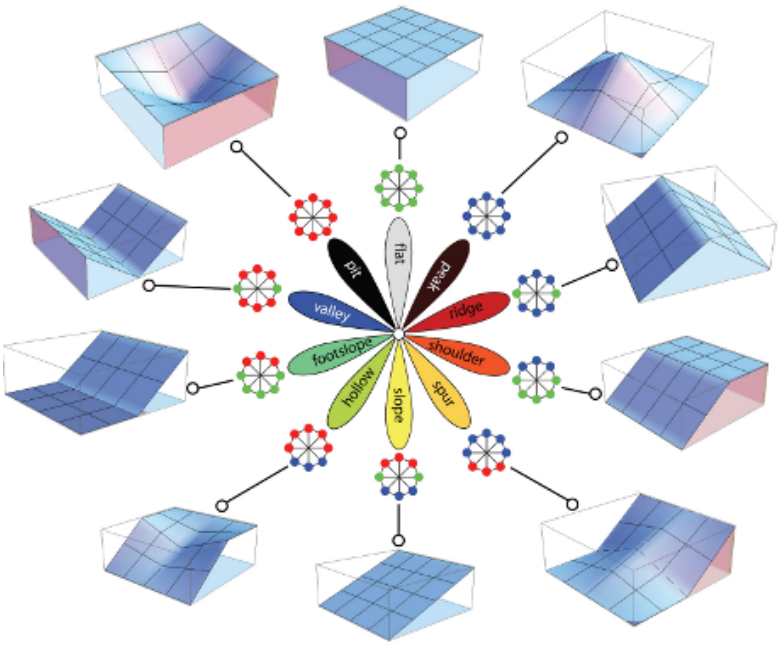

Figure 3.17: Visual representation of terrain classes generated by Grass GIS geomorphon algorithm. Retrieved from [19]

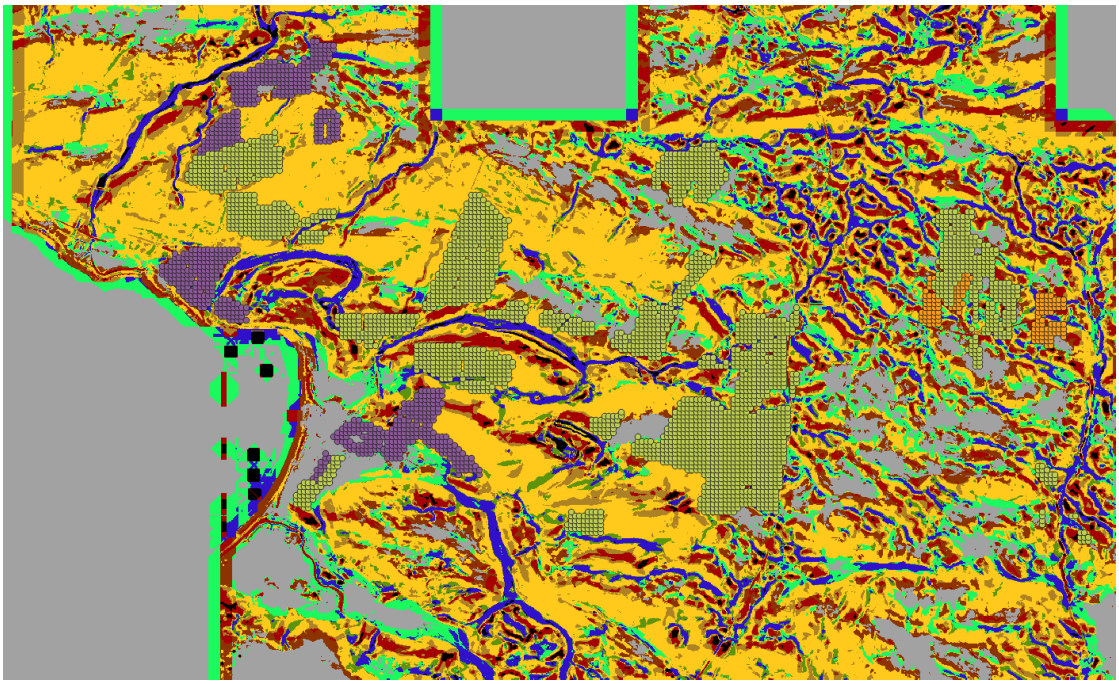

Figure 3.18: Zoomed in portion of the classified terrain with harvest data points on top. The color code is the same as presented in figure 3.17. Notice the very fine resolution of the classification compared to our data grid. The straight green lines at the borders represent the boundaries of our fields of interest.

3.5 Resolutions

When creating the data sets we had to take the different resolutions of the data into account. For the 50 m grid, each entry was based upon the individual coordinates of the harvest data, see Fig. 3.12, which had a lower resolution (50 x 50 m^2) compared to Sentinel-1 (11 x 11 m^2), topology (10 x 10 m^2) and elevation data (10 x 10 m^2). A problem arises due to the resolutions mismatch: The coordinate-wise harvest data summarizes the harvest within a 50 x 50 m^2 area but the finer resolution data can only cover an area of roughly 10 x 10 m^2, see Fig. 3.19. If feature vectors were constructed with only one sample closest in space from the satellite, topology, and elevation data, the whole area that the harvest data summarized would not be covered.

For the 22 m grid, we do not have to take resolutions into account since including any auxiliary data from the neighbors would create a data leakage. This issue is described more in the next sections.

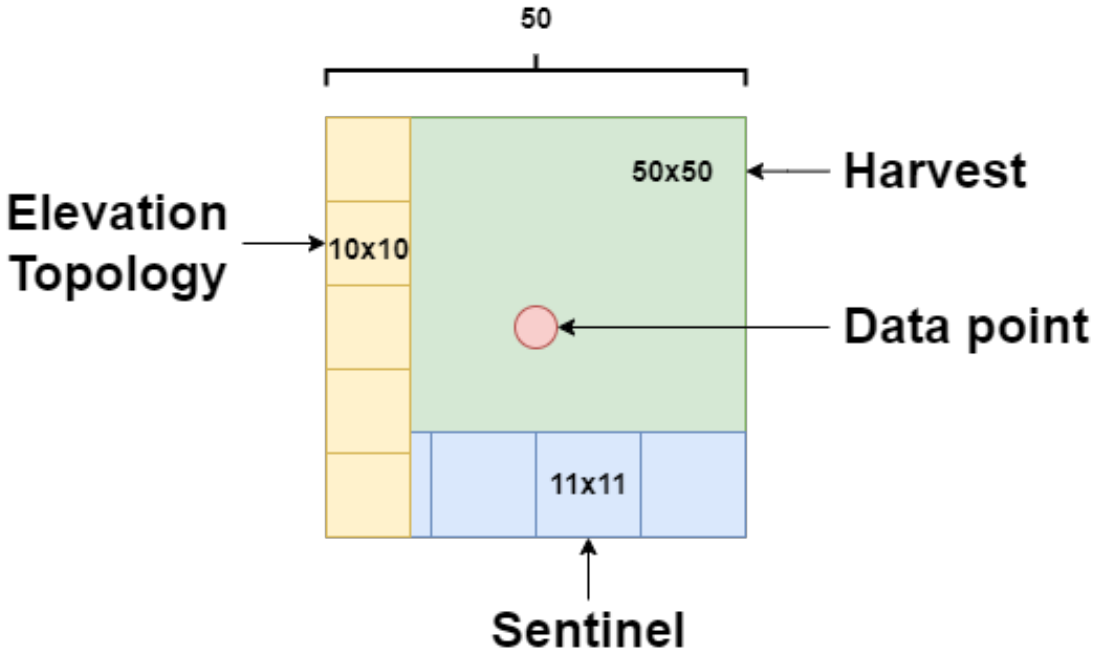

Figure 3.19: Visualization of the mismatch in resolution between the harvest, Sentinel-1, topology, and elevation data. The lowest resolution was the harvest data, where each data point summarizes the harvest within a 50 x 50 m^2 area. The satellite data had a finer resolution of 11 x 11 m^2 whilst the topology and elevation data shared a resolution of 10 x 10 m^2. This resolution difference was the reason for introducing grid sampling instead of just sampling the nearest data point in space.

3.6 Nearest & Grid Sampling

To handle the mismatching resolutions two different samplings approaches were adopted when creating the feature vectors: *Nearest* and *Grid* sampling. For a given coordinates longitude and latitude, Nearest sampling simply ignored the resolution mismatch and picked a single data point nearest in space for Sentinel-1, topology, and elevation, hence the name. For Grid sampling, a neighborhood of data points was picked to form a 3-by-3 grid. The data point closest in space was the center point. From the center point, all neighboring data points in directions North, North-East, East, South-East, South, South-West, West, and North-West (8 additional data points) were added.

Avoidance of data leakage is crucial when grid sampling, to not skew the results. In ML, data leakage happens when information about the test set is available in the training process, leading to a leak of information and giving the algorithm an unfair advantage during testing. Data leakage may result in an overestimation of the algorithm's performance. A data set should in most machine learning tasks be split into different sets, where the training data is the only data the algorithm will see. The rest of the data should not be used for training, nor should it be included in the preprocessing of the data. The data which is not part of the training set is commonly split into a validation set (used for model tuning) and a test set (used for final model evaluation) [55]. Fig. 3.20 illustrates why 3-by-3 Grid sampling does not introduce data leakage for the 50 m grid. When we are using the 22 m grid we only use Nearest sampling since Grid sampling would introduce significant data leakage.

3.7 Machine Learning

Now we have described how all data was collected and preprocessed, the next step is to vectorize all data and use it to train a machine learning model. To do this we have to construct feature vectors, select models, and decide which metrics we want to use to evaluate the performance of the models.

3.7.1 Light Gradient Boosting Machine

Light Gradient Boosting Machine (LGBM) is a machine learning model based on decision trees that belongs to a family of Gradient boosting Decision Tree (GBDT) models [22]. LGBM utilizes Gradient-based One-Side Sampling (GOSS) and Exclusive Feature bundling (EFB). With GOSS, the model learns to exclude a significant proportion of data with small gradients which is a particularly useful aid when the feature importance is unknown. EFB bundles mutually exclusive

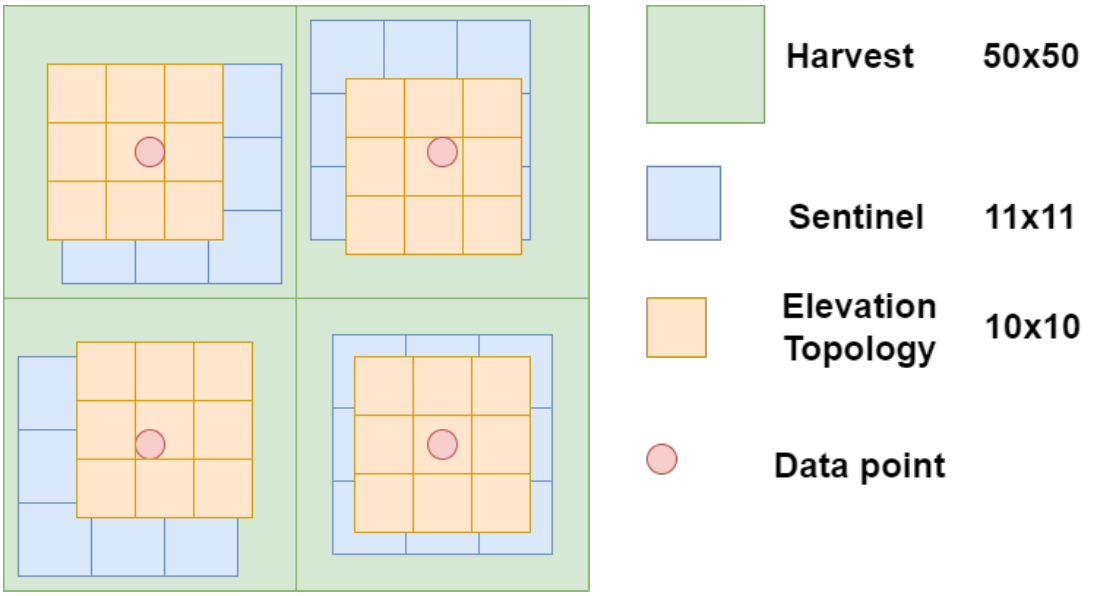

Figure 3.20: Illustration depicting the different cases of sampling data at different resolutions with a grid. The image depicts both good cases (bottom right) and worst cases of grid sampling. Notice how it is impossible for us to get data from the next harvest square with this method. In an edge case scenario, we can get data $\sim$ 22 m from the data point but we can never cross the boundary (25 m) to the next harvest square.

features together to decrease the number of features which speeds up the training process with almost no loss in accuracy. The default hyperparameters were used unless explicitly stated otherwise, which can be found in great detail in the official documentation [32].

We use the LGBM model in this project. The motivation for this is the fact that GBDT models have previously shown good results for harvest prediction tasks with optical data from Sentinel-2 [1]. Another great property of decision trees for this problem is their ability to deal with imbalanced data. Our features are quite imbalanced. Some features such as elevation do not change with time and thus require fewer features to be represented than the satellite data or the weather which varies in time.

3.7.2 Feedforward Neural Network

Feedforward Neural Network (FNN) is the quintessential deep learning model mimicking a network of neurons in the brain (hence *neural network*) by a cyclic directed graph of nodes. Each node, or neuron, have a defined input and output. The input is the feature vector or potentially the output of other nodes in the network. The output of a node is produced by first applying a linear operation on the inputs reducing the vector input to a scalar and then passing it to a non-linear activation

function. These models are called feedforward because the information in the network flows in a forward direction in the network, which is to say that the output of one node is never fed back to that node. As information flows forward, each level of nodes can be viewed as a layer, with an input layer, several hidden layers, and an output layer for the whole network [13]. The depth of a network is measured by the amount of hidden and output layers it contains. The number of nodes in a layer is the layer's width.

When designing an FNN, the depth and width of the hidden layers, activation functions, the desired output type, learning algorithm, and regularization need to be considered [13, 43, 38, 56]. We decided to have a depth of 6 and a scaling width for each hidden layer depending on the number of features, n_f, in the input with a certain minimum or maximum width for each layer:

- Hidden Layer 1 width := $\max(5 \cdot n_f, 500)$
- Hidden Layer 2 width := $\max(3 \cdot n_f, 300)$
- Hidden Layer 3 width := $\max(2 \cdot n_f, 200)$
- Hidden Layer 4 width := $\min(\max(0.5 \cdot n_f, 50), 200)$
- Hidden Layer 5 width := $\min(\max(0.2 \cdot n_f, 20), 50)$

Each hidden layer had a Rectified Linear Unit (ReLU) activation function. The output layer was always a single node with linear activation, as this was a regression problem. For our gradient descent optimizer, we picked Adam with default hyperparameters [23], as tests have shown that this algorithm is a good default optimizer with adaptive learning rates [38]. The loss was the mean squared error. A maximum of 50 epochs were used, with an Early Stopping scheme monitoring the validation loss with 10 epochs of patience used to regularize and stop overfitting. The validation set was randomly picked 10 % of the training set and the batch size was 32.

3.7.3 Metrics

To evaluate our machine learning models we use Root Mean Squared Error (RMSE), Accuracy (Acc.) and weighted F1-score (F1) [40]. The definitions of these metrics are shown below. Let y_i be the ground truth measurement from the harvest data, $\hat{y}_i$ be the predicted value, TP/FP be True/False positive, and TN/FN be True/False negative.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \qquad \text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

$$\text{Precision} = \frac{TP}{TP + FP} \qquad \text{F1-score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

The metrics are both for regression and classification tasks [35]. A benefit of using RMSE is that the regression error will have the same unit as the labels, i.e. average measured harvest in tonnes per hectare. Harvest prediction is a regression problem but can be reformulated as a classification problem by rounding harvest values [1]. This allows us to easily compare how Sentinel-1 performs against Sentinel-2 for harvest prediction for winter wheat fields in the same region.

A significant problem with rounding to the nearest integer to create classes is that close prediction in harvest might yield different classes and vice versa. For example, say the true harvest was 6.51 (rounded to 7) tonnes and that the predicted harvest was 6.49 (rounded to 6). The prediction would then be very close and an excellent estimation, but still be counted as an incorrect classification. In contrast, an objectively worse prediction, say 7.49 (rounded to 7) would be counted as a correct classification, even though significantly larger error. This issue is due to the task of harvest prediction fundamentally being a regression task and not a classification task. Hence, just rounding to the nearest integer skews the results and should not be used.

A better way of rounding would be to take the RMSE into account. For instance, we want RMSE predictions that are within some error tolerance ϵ to be assigned the same class as the true harvest. To do this, we start by creating our classes. We will use the non-negative integers as our classes, i.e. 0, 1, 2, etc., which is what the authors did in the study using Sentinel-2 [1]. Now, given that the true harvest and predicted harvest are within some error, we want both to belong to the same class. Let us illustrate this with an example. Given that the error tolerance is $\epsilon = 0.5$, the true harvest is 10.7 and the predicted harvest is 10.3, we see that the absolute value of the difference will be $|10.7 - 10.3| = 0.4 < 0.5 = \epsilon$. Because the value is within the error tolerance, this will be a correct classification. So, how should we round to make this the case? If we blindly rounded to the nearest integer we would have true harvest 11 and predicted harvest 10. We instead do the following: If the prediction is within the tolerance, set the prediction to the same class as the true harvest regardless of rounding, i.e. 10.3 is set to 11 in our

example. Note, for the true harvest we always do integer rounding. It is just the predicted harvest that needs extra attention. Another example, given that the error tolerance is $\epsilon = 0.5$, the true harvest is 10.7 and the predicted is 11.4, we see that the absolute value of the difference will be $|10.7 - 11.4| = 0.7 > 0.5 = \epsilon$. The value is not within the error tolerance and as such, the rounding needs to be done so they do not share the same class. Here, we see another issue that needs special attention. If we just rounded both numbers independently then we would get 11 as the true harvest class and predicted harvest class. Therefore, we added extra conditions in our code to handle this. The code checked first if the prediction was within the error tolerance, and if it was not then we rounded both independently to classes. If the prediction was not within the error tolerance and the true and predicted harvest were rounded to the same class, then the predicted harvest was changed to a class above or below the true harvest. In the previous example, because the predicted harvest was $11.4 > 10.7$ the class was changed from 11 to 12. If the predicted harvest was lower, then the class would instead have been 10. Note, that this alters the notion of a clearly defined class but is what we found to be the best way to handle the issues by just rounding to the nearest integer. The full code for transforming RMSE to classes can be seen in Appendix B.2. Unless otherwise stated, we will use the same default value of $\epsilon = 0.5$ as authors in [1] for our classification results.

3.7.4 K-Fold Cross-Validation

K-fold cross-validation is a common way of evaluating model performance, and it does this by dividing the data set into k folds. Each fold contains a random and distinct split of the data: a training set and a test set. The model is then trained with the training set and tested with the test set (unseen data). The metrics are averaged over all folds, giving a mean of the performance with k different splits [39, 54]. By averaging over k different splits, a better performance evaluation of the model is achieved.

3.7.5 Feature Importance

Feature importance (FI) algorithms can be used to reduce the number of features in a data set before training a model. This is particularly useful when we do not know which features are useful and which are not. This property allows us to use feature importance algorithms to qualitatively make sense of and explain a complex problem with many parameters. Since feature importance algorithms reduce the number of features in a data set, it reduces training speed as a consequence. Sometimes, in best-case scenarios, it can also increase the performance of a model

by dropping uncorrelated features which essentially just become noise in a model.

The feature importance algorithm we chose to use was Boruta [26]. Boruta can be thought of as an extension of a Random Forest model. It tests random combinations of features and samples, with a random forest classification test, and keeps the ones that throughout several trials show a consistent correlation with the target data.

We used the python package `Boruta`, which supplies an implementation of the Boruta algorithm [4]. We used a Random Forest model from scikit-learn [41] with balanced class weights and a test size of 20 % as an estimator for the Boruta algorithm. A maximum of 100 iterations were used, with the rest of the hyperparameters set to default. The result from the execution of the algorithm is two lists of features: support and weak support. We decided to use the features in both of these lists. Furthermore, the execution of the algorithm was repeated 17 times and the two lists from each individual run were merged with one final list. This was done separately for both Grid sampling and Nearest sampling of the 50 m grid. The final list of merged features from the 17 runs represents the final feature importance vector.

3.7.6 Transfer Learning

Transfer learning means reusing a pre-trained model on a new problem for which it was not initially trained for [7]. In the context of harvest prediction, this means training a model on data from one year and testing its performance on data from another year. Achieving successful transfer learning is the holy grail of modeling harvest prediction. It would be a major step towards operational use since it could be used to predict the harvest when it is still possible for the farmers to affect the outcome.

We test how well our models do in transfer learning by training them with data from one year and testing on data from all other years in our data set. To make this a fair test, we randomly pick samples from the test year such that the test size corresponds to 10% of the training size, just as we did when we tested each year on itself.

3.7.7 Feature Vectors

A number of different feature vectors Table 3.1 were used to test how features independently and joined affect the ML algorithms. For the main time period spanning week 14 to week 29, different features were either sampled on a weekly basis or once for the whole time period. The features VH & VV and Weather were sampled once per week. For a given week, there may be a multitude of VH &

Table 3.1: List of our final feature sets and their sizes.

Feature type	Description	#Features (Nearest)	#Features (Grid)
VH & VV	Processed and min-max normalized VH and VV bands	32	288
Ratios	Processed and min-max normalized VH/VV and VV/VH	32	288
Indices	Normalized indices computed with VH and VV	64	576
Weather	Weekly accumulated or averaged temperature, solar radiation intensity, precipitation, wind speed and humidity percentage.	85	-
Topology	One hot encoded topological classification of terrain	10	90
Elevation	Min-max normalized elevation above sea level	1	9
All	Combining VH & VV, Weather, Topology and Elevation	128	472
FI	Filtering of the most important features in All by the Boruta feature selection algorithm	Varies	Varies

VV samples from different dates. Only one of the dates in the week was chosen, with the criteria being that the bands are defined, i.e. not NaN. Topology and Elevation were not sampled once per week in the main time period - they were simply added once for each coordinate.

Chapter 4

Results

In the following sections results for both harvest data with a resolution of 50×50 m^2 and 22×22 m^2 are shown. All sections before Section 4.2 use harvest data with a resolution of 50×50 m^2 and all parts after contain results using harvest data with a resolution of 22×22 m^2.

4.1 Harvest Data With 50×50 m^2 Resolution

4.1.1 Harvest Prediction Using Exclusively Sentinel-1 Data

One would expect that the performance of a machine learning model would get better and better with more data. This is a fundamental assumption and can be thought of as the data having a pattern that the model can learn. If this is not the case, it would suggest that there is nothing in the data which algorithms can learn or that the data contains other errors, e.g. RFI.

To test this assumption, a K-fold cross-validation scheme was created with the model being LGBM evaluated with RMSE metric over an increasing size of the data sets. The harvest data with a resolution of 50×50 m^2 was used for ground truth labels. Put simply, given the assumption above, we expect that the error will decrease as the size of the data set increases. In the tests, $k = 5$ was picked resulting in 80 % of the data for training and 20 % for testing. The LGBM model with 2000 boosted trees and the rest as default hyperparameters [32] was picked as it has shown to be a powerful model for harvest prediction [1] and the training time was short (below 5 seconds using NVIDIA T4 Tensor Core GPU). For the tests, an Early Stopping scheme with patience (`stopping_rounds`) of 200 boosting iterations was used to prevent overfitting,

The results using different feature vectors based upon only Sentinel-1 data are shown in Fig. 4.1, Fig. 4.2 and Fig. 4.3. For all figures, a clear downward trend in the cross-validated RMSE is seen, which implies that there is a pattern

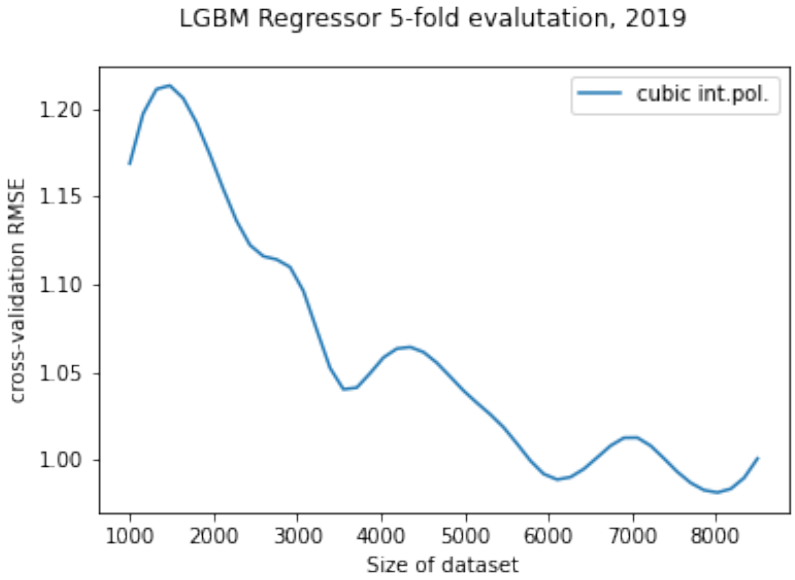

Figure 4.1: The regression RMSE over the size of the data set for the year 2019, using Grid sampled and normalized VH and VV features for each coordinate in the harvest data with a resolution of 50×50 m^2. A clear downward trend in error as the size of the data set increases, implying that the time series of Sentinel-1 data contains valuable information for harvest prediction.

that the algorithm can learn. Interestingly, there is no significant difference in the performance of the algorithm with different derivatives of VH and VV. The simplest version, using only the normalized VH and VV, results in the lowest recorded RMSE error of 0.9813 (Fig. 4.1) compared with the more complex feature vectors containing additional ratios and indices, with errors 1.0037 and 0.9905 respectively (Fig. 4.2, Fig. 4.3) for 2019 data.

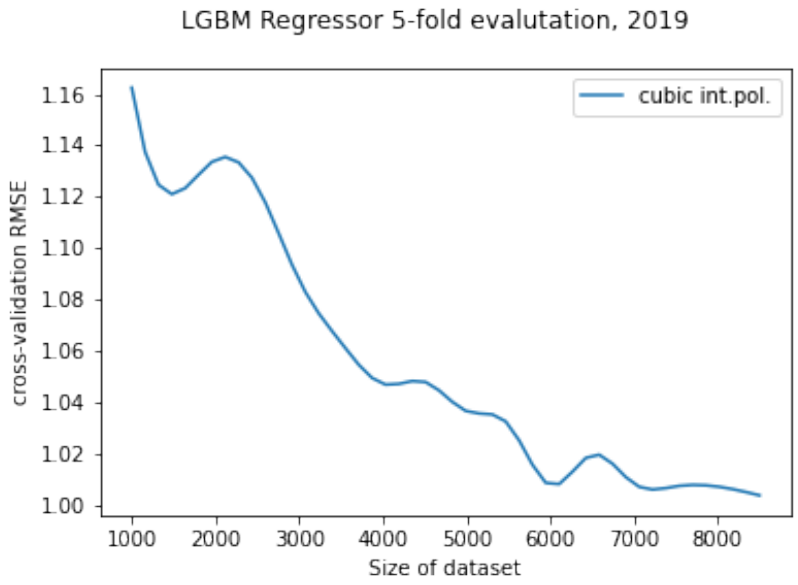

Figure 4.2: The regression RMSE over the size of the data set for the year 2019 using Grid sampled and normalized VH, VV, and ratios features for each coordinate in the harvest data with a resolution of 50×50 m^2. A clear downward trend in error as the size of the data set increases, implying that the time series of Sentinel-1 data contains valuable information for harvest prediction.

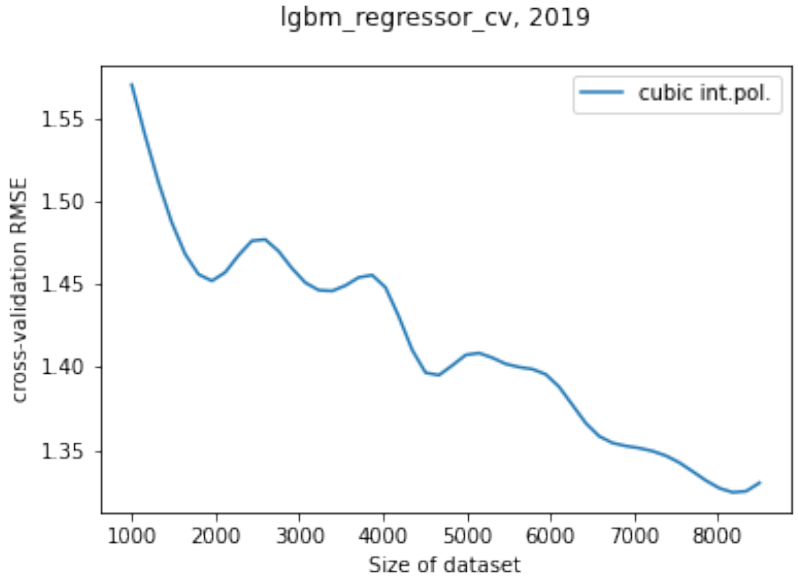

Figure 4.3: The regression RMSE over the size of the data set for the year 2019 using Grid sampled and normalized VH, VV, ratios, and indices features for each coordinate in the harvest data with a resolution of 50×50 m^2. A clear downward trend in error as the size of the data set increases, implying that the time series of Sentinel-1 data contains valuable information for harvest prediction.

Table 4.1: The top 5 important features from the Boruta feature selection algorithm when using Grid sampling and harvest data with a resolution of 50×50 m^2 for each year. The main features were VH, VV, accumulated rain, and elevation (named height for 2018). Ratios, indices, and topology were not in the top 5 features for any of the years.

	2017	2018	2019	2020
1.	se_vv_w_23	w_30_acc_rain_sum	c_vh_w_21	w_27_acc_rain_sum
2.	e_vh_w_24	w_21_acc_rain_sum	n_vh_w_21	e_vh_w_21
3.	s_vv_w_23	w_height	ne_vh_w_21	ne_vh_w_27
4.	sw_vh_w_24	sw_height	e_vh_w_21	n_vh_w_27
5.	e_vh_w_23	se_height	w_vv_w_22	w_17_acc_rain_sum

4.1.2 Feature Selection

Several features were included in the feature selection process, based upon the harvest data with a resolution of 50×50 m^2, and the top 5 can be seen in Tables 4.1 and 4.2. The naming of the features is based on several reasons: position in the neighbor grid, week number, and type of feature. The features which have a position abbreviation prefix are VH & VV, topology and elevation, e.g. "c" - center, "ne" - North-East, etc. The week number was either prefixed or appended to the end of the feature name with a "w" followed by an underscore and then the week number. Each feature type had its unique string, e.g. "vv" - VV, "vh" - VH, "acc_rain_sum" - accumulated rain, "height" - elevation, etc.

The VH and VV bands were generally part of the top 5 features for both Grid and Nearest sampling. Grid sampling seems to favor a more diverse set of features whilst the VH and VV bands dominated the important feature rankings for Nearest sampling. Please contact the authors for a complete list of the rankings and features selected.

4.1.3 General Results

The general results using harvest data with a resolution of 50×50 m^2 contain one table for each year which consists of all performance metrics relevant to that year using both sampling methods (Grid and Nearest), both models (LGBM and FNN), and all main sets of input features, see Tables 4.3, 4.4, 4.5 and 4.6. To get the classification score from RMSE $\epsilon = 0.5$ was used. Ratios and indices were not part of any of the sets, as the results in Section 4.1.1 showed no difference in the performance compared to just the VH & VV bands.

There was no clear significant difference in performance between the sampling methods, with most results being almost identical or slight performance difference

Table 4.2: The top 5 important features from the Boruta feature selection algorithm when using Nearest sampling and harvest data with a resolution of 50×50 m^2 for each year. VH and VV bands dominated the important features, with elevation being the only other feature type present in the top 5 for 2018. It is interesting that with Nearest sampling none of the years contains any accumulated rain feature, which was an important feature when Grid sampling (Table 4.1). Weather, ratios, indices, and topology were not in the top 5 features for any of the years.

	2017	2018	2019	2020
1.	c_vh_w_23	c_height	c_vh_w_18	c_vv_w_20
2.	c_vv_w_23	c_vh_w_21	c_vh_w_19	c_vh_w_21
3.	c_vh_w_24	c_vv_w_22	c_vh_w_20	c_vh_w_27
4.	c_vh_w_20	c_vv_w_26	c_vh_w_21	c_vv_w_27
5.	c_vv_w_19	c_vh_w_22	c_vv_w_22	c_vv_w_28

(Tables 4.3, 4.4, 4.5 and 4.6). A benefit of using Nearest sampling over Grid sampling is the reduction in features leading to significantly faster training and evaluation times.

For all years, sampling methods, and feature vectors LGBM outperformed the FNN model (Tables 4.3, 4.4, 4.5 and 4.6). Another significant benefit of the LGBM model was the significantly faster K-fold evaluation ($\sim$32x faster).

4.1.4 Harvest Distributions

It was interesting to compare the true harvest distribution with the predicted distributions from LGBM, our best-performing model, for all years using harvest data with a resolution of 50×50 m^2. The true distribution (left) and predicted distribution (right) for each year can be seen in Figures 4.4, 4.5, 4.6 and 4.7. In general, the predicted distribution was similar to the true distribution, having roughly the same shape, mean, and variance.

4.1.5 Varying Error Margin for Classifiers

This section investigates how the classification and F1 scores change when we increase the error tolerance ϵ described in Section 3.7.3. This provides an alternative way to interpret the results instead of just looking at the RMSE. Table 4.7 shows the accuracy and F1 scores when we vary ϵ from 0.50 to 2.00 tonnes per hectare.

Table 4.3: Summary of results for 2017 using harvest data with a resolution of 50×50 m^2 with the different sampling methods, feature vectors, and models. The different sampling methods yielded similar results, with no clear benefit of one over the other. The best overall performance was achieved with LGBM using the All feature vector, independent of the sampling method. FI and VH & VV were not far away from having the same scores. Looking at feature vectors independently and not combined, VH & VV was the best performing feature vector with Weather, Topology, and Elevation having significantly worse performance. VH & VV and Weather had much more features compared to Topology and Elevation, which should be taken into consideration.

	LGBM			FNN		
Features, Grid	RMSE	Acc.	F1	RMSE	Acc.	F1
All	0.80	0.56	0.55	1.09	0.41	0.40
Feature importance	0.85	0.53	0.52	1.12	0.44	0.41
VH & VV	0.85	0.54	0.52	1.12	0.44	0.41
Topology	1.35	0.34	0.28	1.47	0.33	0.28
Elevation	1.23	0.37	0.34	1.33	0.37	0.30

	LGBM			FNN		
Features, Nearest	RMSE	Acc.	F1	RMSE	Acc.	F1
All	0.81	0.56	0.55	1.05	0.43	0.41
Feature importance	0.85	0.55	0.54	1.18	0.44	0.41
VH & VV	0.86	0.53	0.52	1.12	0.45	0.43
Weather	1.08	0.41	0.38	1.29	0.36	0.32
Topology	1.32	0.36	0.27	1.40	0.34	0.26
Elevation	1.24	0.36	0.32	1.33	0.35	0.25

Table 4.4: Summary of results for 2018 using harvest data with a resolution of 50×50 m^2 with the different sampling methods, feature vectors, and models. The different sampling methods yielded similar results, with some significant differences, e.g. FI for both LGBM and FNN. The best overall performance was achieved with LGBM using the All feature vector, independent of the sampling method. This score was also the best generally for all years tested. Looking at feature vectors independently and not combined, VH & VV was the best performing feature vector with Weather, Topology, and Elevation having significantly worse performance. VH & VV and Weather had much more features compared to Topology and Elevation, which should be taken into consideration.

	LGBM			FNN		
Features, Grid	RMSE	Acc.	F1	RMSE	Acc.	F1
All	0.75	0.60	0.59	0.87	0.50	0.49
Feature importance	0.92	0.49	0.48	1.70	0.38	0.36
VH & VV	0.81	0.58	0.58	0.87	0.50	0.49
Topology	1.51	0.31	0.24	1.58	0.28	0.22
Elevation	1.26	0.38	0.35	1.36	0.33	0.26

	LGBM			FNN		
Features, Nearest	RMSE	Acc.	F1	RMSE	Acc.	F1
All	0.76	0.60	0.59	0.87	0.48	0.47
Feature importance	0.78	0.58	0.57	1.25	0.49	0.47
VH & VV	0.81	0.58	0.57	0.88	0.50	0.49
Weather	1.11	0.39	0.32	1.39	0.33	0.27
Topology	1.49	0.32	0.22	1.57	0.29	0.21
Elevation	1.29	0.36	0.32	1.35	0.35	0.28

Table 4.5: Summary of results for 2019 using harvest data with a resolution of 50×50 m^2 with the different sampling methods, feature vectors, and models. The different sampling methods yielded similar results, with some minor differences, e.g. All for FNN. The best overall performance was achieved with LGBM using the All feature vector, independent of the sampling method. Looking at feature vectors independently and not combined, VH & VV was the best performing feature vector with Weather, Topology, and Elevation having significantly worse performance. VH & VV and Weather had much more features compared to Topology and Elevation, which should be taken into consideration.

	LGBM			FNN		
Features, Grid	RMSE	Acc.	F1	RMSE	Acc.	F1
All	0.93	0.50	0.50	1.07	0.41	0.41
Feature importance	0.99	0.48	0.47	1.13	0.37	0.36
VH & VV	1.01	0.48	0.47	1.15	0.41	0.40
Topology	1.82	0.19	0.12	1.90	0.19	0.14
Elevation	1.68	0.23	0.19	1.85	0.18	0.10

	LGBM			FNN		
Features, Nearest	RMSE	Acc.	F1	RMSE	Acc.	F1
All	0.93	0.49	0.49	1.13	0.40	0.39
Feature importance	0.97	0.48	0.48	1.12	0.40	0.40
VH & VV	1.04	0.47	0.46	1.12	0.41	0.40
Weather	1.34	0.31	0.29	1.52	0.26	0.23
Topology	1.80	0.19	0.10	1.84	0.18	0.10
Elevation	1.75	0.21	0.16	1.83	0.18	0.09

Table 4.6: Summary of results for 2020 using harvest data with a resolution of 50×50 m^2 with the different sampling methods, feature vectors, and models. The different sampling methods yielded similar results, with some significant differences, e.g. FI for both LGBM and FNN. The best overall performance was achieved with LGBM using either the All feature vector with Grid sampling or using the FI feature vector with Nearest sampling. Looking at feature vectors independently and not combined, VH & VV was the best performing feature vector with Weather, Topology, and Elevation having significantly worse performance. VH & VV and Weather had much more features compared to Topology and Elevation, which should be taken into consideration.

	LGBM			FNN		
Features, Grid	RMSE	Acc.	F1	RMSE	Acc.	F1
All	1.08	0.44	0.44	1.37	0.31	0.30
Feature importance	1.21	0.40	0.40	1.64	0.31	0.31
VH & VV	1.20	0.40	0.40	1.47	0.36	0.36
Topology	2.67	0.18	0.13	2.77	0.16	0.12
Elevation	2.45	0.21	0.18	2.70	0.17	0.11

	LGBM			FNN		
Features, Nearest	RMSE	Acc.	F1	RMSE	Acc.	F1
All	1.09	0.44	0.43	1.37	0.36	0.35
Feature importance	1.08	0.43	0.43	1.33	0.35	0.34
VH & VV	1.25	0.38	0.38	1.53	0.32	0.32
Weather	1.66	0.25	0.23	2.05	0.23	0.22
Topology	2.64	0.19	0.11	2.67	0.19	0.12
Elevation	2.54	0.21	0.16	2.68	0.18	0.11

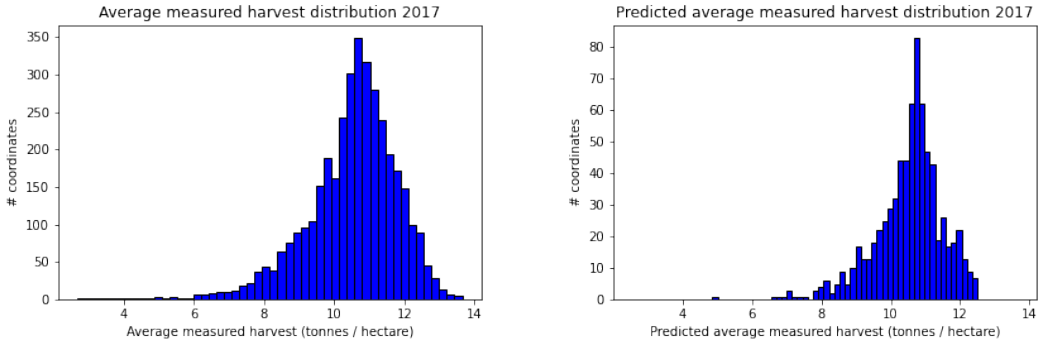

Figure 4.4: True distribution (left) and predicted distribution (right) for 2017 using harvest data with a resolution of 50×50 m^2. The predicted distributions are from an LGBM model with the All feature vector, which had among the best overall performance for all years, see Section 4.1.3. The model accurately captured the true distribution for this year, having roughly the same shape, mean, and variance. The model seemed to have had an issue with predicting large harvest, which can be seen in the lack of predicted harvest of about 13-14 tonnes per hectare. However, this might be due to the lower sample size of the test set.

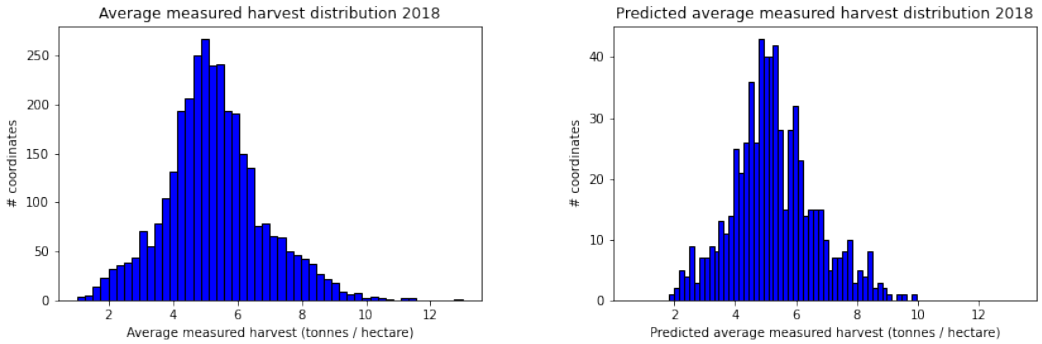

Figure 4.5: True distribution (left) and predicted distribution (right) for 2018 using harvest data with a resolution of 50×50 m^2. The predicted distributions are from an LGBM model with the All feature vector, which had among the best overall performance for all years, see Section 4.1.3. The model accurately captured the true distribution for this year, having roughly the same shape, mean, and variance.

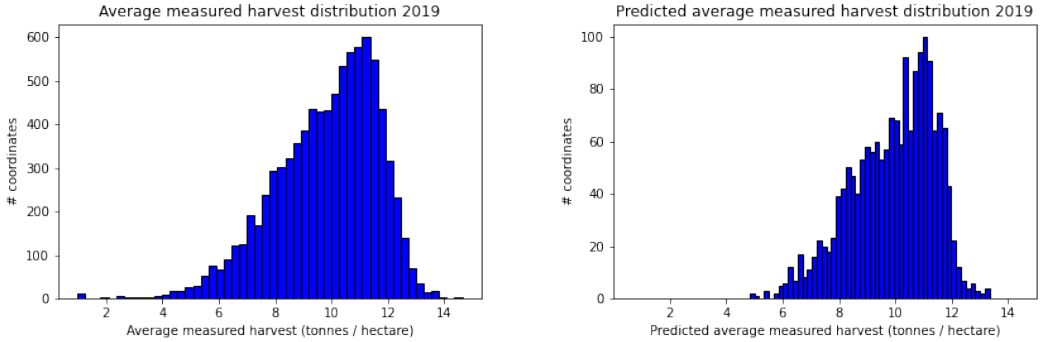

Figure 4.6: True distribution (left) and predicted distribution (right) for 2019 using harvest data with a resolution of 50×50 m^2. The predicted distributions are from an LGBM model with the All feature vector, which had among the best overall performance for all years, see Section 4.1.3. The model accurately captured the true distribution for this year, having roughly the same shape, mean, and variance.

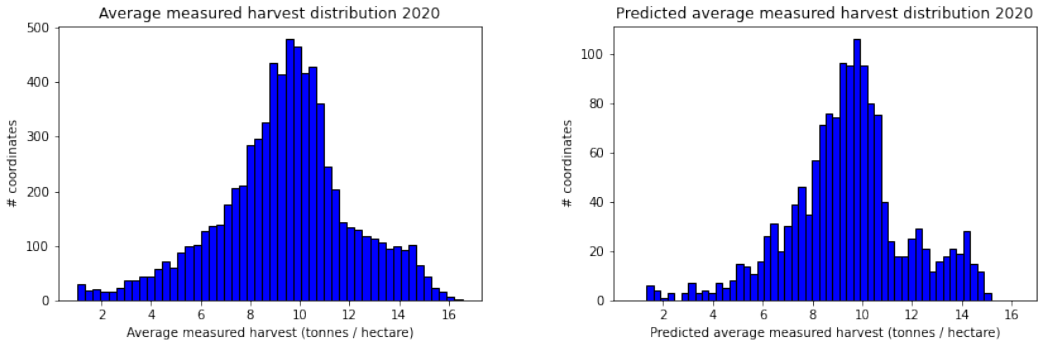

Figure 4.7: True distribution (left) and predicted distribution (right) for 2020 using harvest data with a resolution of 50×50 m^2. The predicted distributions are from an LGBM model with the All feature vector, which had among the best overall performance for all years, see Section 4.1.3. The model accurately captured the true distribution for this year, having roughly the same shape, mean, and variance.

Table 4.7: The varying error tolerance for the LGBM regressor trained with the All feature vector, Grid sampling, and harvest data with a resolution of 50×50 m^2. The error tolerance, ϵ, varies from 0.50 to 2.00 tonnes per hectare. Note that the performance increases significantly between $\epsilon \in [0.5, 1.0]$.

	2017		2018		2019		2020	
Error tolerance, ϵ	Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1
0.50	0.56	0.55	0.60	0.59	0.50	0.50	0.44	0.44
0.75	0.72	0.71	0.76	0.76	0.67	0.67	0.60	0.60
1.00	0.83	0.82	0.86	0.86	0.79	0.78	0.72	0.72
1.25	0.90	0.89	0.91	0.91	0.86	0.86	0.80	0.79
1.50	0.94	0.93	0.95	0.94	0.91	0.91	0.86	0.86
1.75	0.96	0.95	0.97	0.96	0.94	0.94	0.90	0.90
2.00	0.97	0.97	0.98	0.98	0.96	0.96	0.93	0.93

4.1.6 Transfer Learning

Table 4.8 presents the results of the transfer learning using harvest data with a resolution of 50×50 m^2. The performance is very low, which should not come as a surprise since the distributions of harvest between the years varies significantly (Section 4.1.4). The best overall performance was achieved by training the model with data from 2020 and then using that model to predict harvest on the data from 2017.

4.1.7 Effects of RFI Filtering

The occurrence of RFI in the data might hamper the performance of the models and therefore the performance of the models was investigated using an RFI-filtered version of the Sentinel-1 data. The results can be seen in Table 4.9. The proposed RFI filtering algorithm significantly reduced the number of data points for each year. For 2018, the decrease was so large that the total number of data points was below 100 after filtering. As such, training and testing for 2018 were no longer feasible.

4.2 Harvest Data With 22×22 m^2 Resolution

4.2.1 Harvest Prediction Using Exclusively Sentinel-1 Data

A finer grid for the harvest data (22×22 m^2) results in a much larger data set (more than 4x in size) and the effect of the increase in size was tested with the

Table 4.8: A summary over transfer learning with LGBM trained with the All feature vector, Grid sampling, and harvest data with a resolution of 50×50 m^2 from one year and tested on data from another year. The true distributions (Section 4.1.4) varied over the years, implying transfer learning should be difficult, which can be seen in the large RMSE error and low classification scores. The best overall score was with the model being trained with data from 2020 and the test set from 2017. Comparing the distributions for years 2017 and 2020 (Figure 4.4) and 4.7) some general similarity can be seen and might be the reason for the better performance. The distribution for 2018, Fig. 4.5, had a significantly lower mean compared to the other years, and this might also be the reason for the low performance when trying to predict 2018 with a model trained with data from another year.

training year	test year	RMSE.	Acc.	F1
2017	2018	4.44	0.02	0.01
2017	2019	1.73	0.19	0.17
2017	2020	2.30	0.16	0.14
2018	2017	3.91	0.03	0.02
2018	2019	3.07	0.07	0.06
2018	2020	3.14	0.09	0.05
2019	2017	1.99	0.16	0.16
2019	2018	3.53	0.03	0.01
2019	2020	2.04	0.18	0.17
2020	2017	1.69	0.22	0.23
2020	2018	3.75	0.02	0.01
2020	2019	2.16	0.18	0.19

Table 4.9: Comparison between data with or without RFI filter. For our setup and the proposed RFI filter we see no improvements in the metrics, but a slight decrease in performance across the boards. The model for all tests was LGBM using the All feature vector, Grid sampling, and harvest data with a resolution of 50×50 m^2. The results for 2018 are not included, as the RFI filter removed a significant number of data points resulting in training and testing not yielding any meaningful results.

	RFI filter			No filter		
Year	RMSE	Acc.	F1	RMSE	Acc.	F1
2017	0.86	0.53	0.53	0.80	0.56	0.55
2019	0.95	0.50	0.50	0.93	0.50	0.50
2020	1.11	0.43	0.43	1.08	0.44	0.44

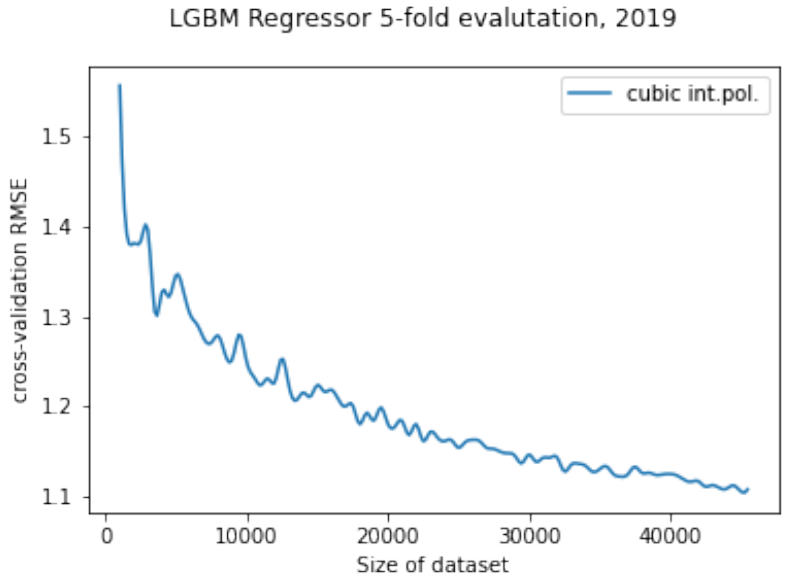

Figure 4.8: The regression RMSE over the size of the data set for the year 2019 using Nearest sampling and normalized VH and VV features for each coordinate in the harvest data with resolution 22 × 22 m^2. A clear downward trend in error as the size of the data set increases, implying that the time series of Sentinel-1 data contains valuable information for harvest prediction. Compared with results using harvest data with resolution 50 × 50 m^2 and Grid sampling (Fig. 4.1), the error was higher for the finer resolution.

same K-fold cross-validation scheme used in Section 4.1.1. The results are shown in Figures 4.8, 4.9 and 4.10. Interestingly, the increase in the size of the data set did not yield any lower RMSE error but instead a higher error, compare Figures 4.1, 4.2 and 4.3 with Figures 4.8, 4.9 and 4.10. This was not expected, as generally more data is better for ML models.

4.2.2 General Results

The general results using harvest data with a resolution of 22 × 22 m^2 are shown in the same way as for 50 × 50 m^2, with the exception being only results for Nearest sampling, see Tables 4.10, 4.11, 4.12 and 4.13. To get the classification score from RMSE $\epsilon = 0.5$ was used. In general, the lower resolution grid of harvest data outperformed (Section 4.1.3) the finer grid. For all years and feature vectors, LGBM outperformed the FNN model (Tables 4.10, 4.11, 4.12 and 4.13).

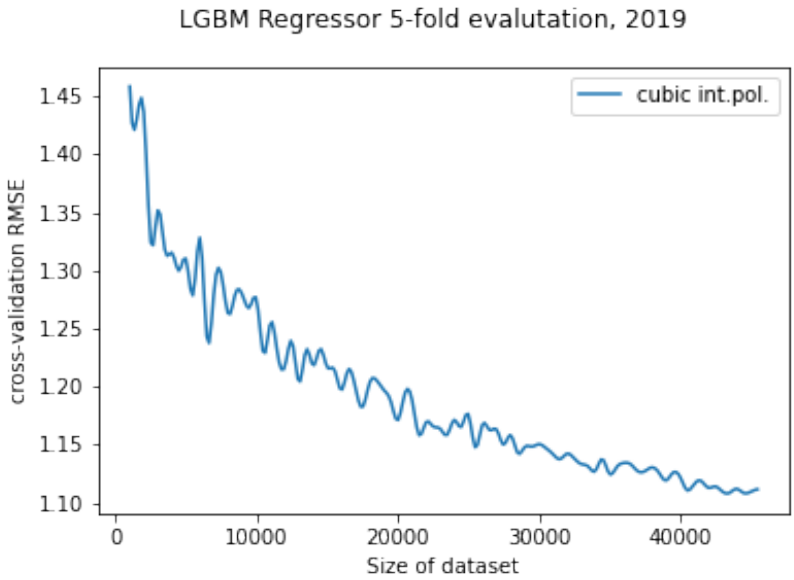

Figure 4.9: The regression RMSE over the size of the data set for the year 2019 using Nearest sampling and normalized VH, VV, and ratios feature for each coordinate in the harvest data with resolution 22×22 m^2. A clear downward trend in error as the size of the data set increases, implying that the time series of Sentinel-1 data contains valuable information for harvest prediction. Compared with results using harvest data with resolution 50×50 m^2 and Grid sampling (Fig. 4.2), the error was significantly higher for the finer resolution.

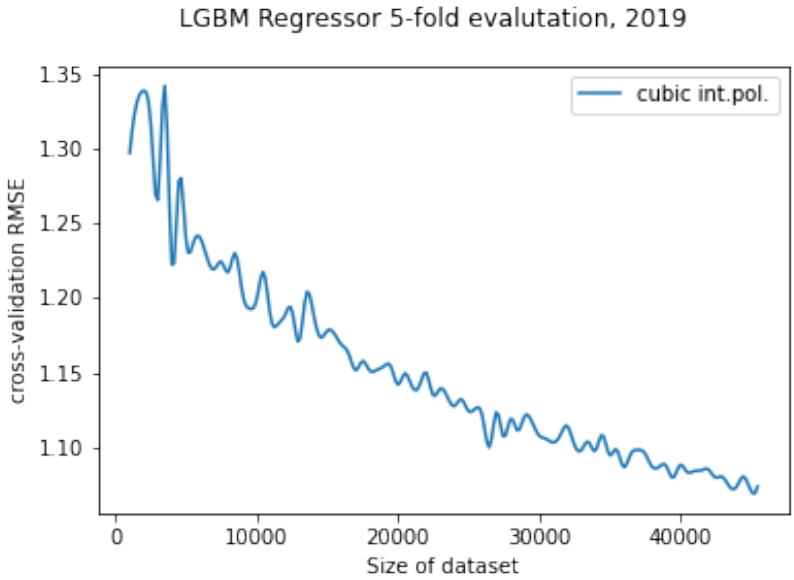

Figure 4.10: The regression RMSE over the size of the data set for the year 2019 using Nearest sampling and normalized VH, VV, ratios, and indices features for each coordinate in the harvest data with resolution 22×22 m^2. A clear downward trend in error as the size of the data set increases, implying that the time series of Sentinel-1 data contains valuable information for harvest prediction. Compared with results using harvest data with resolution 50×50 m^2 and Grid sampling (Fig. 4.3), the error was significantly lower for the finer resolution. This was the only case when the finer grid achieved better RMSE error.

Table 4.10: Summary of results for 2017 using harvest data with a resolution of 22×22 m^2, Nearest sampling, different feature vectors, and models. The best overall performance was achieved with LGBM using the All feature vector. Looking at feature vectors independently and not combined, VH & VV was the best performing feature vector with Weather, Topology, and Elevation having significantly worse performance. VH & VV and Weather had much more features compared to Topology and Elevation, which should be taken into consideration.

	LGBM			FNN		
Features, Nearest	RMSE	Acc.	F1	RMSE	Acc.	F1
All	1.07	0.48	0.47	1.26	0.39	0.38
VH & VV	1.11	0.47	0.45	1.25	0.41	0.39
Weather	1.35	0.37	0.33	1.47	0.36	0.32
Topology	1.64	0.26	0.17	1.67	0.26	0.17
Elevation	1.51	0.30	0.26	1.63	0.27	0.18

Table 4.11: Summary of results for 2018 using harvest data with a resolution of 22×22 m^2, Nearest sampling, different feature vectors, and models The best overall performance was achieved with LGBM using the All feature vector. This score was also the best generally for all years tested with the finer resolution. Looking at feature vectors independently and not combined, VH & VV was the best performing feature vector with Weather, Topology, and Elevation having significantly worse performance. VH & VV and Weather had much more features compared to Topology and Elevation, which should be taken into consideration.

	LGBM			FNN		
Features, Nearest	RMSE	Acc.	F1	RMSE	Acc.	F1
All	0.89	0.54	0.53	1.06	0.44	0.43
VH & VV	0.93	0.52	0.51	1.02	0.47	0.46
Weather	1.32	0.33	0.29	1.37	0.32	0.28
Topology	1.72	0.26	0.18	1.74	0.26	0.16
Elevation	1.52	0.28	0.23	1.57	0.26	0.21

Table 4.12: Summary of results for 2019 using harvest data with a resolution of 22×22 m^2, Nearest sampling, different feature vectors, and models The best overall performance was achieved with LGBM using the All feature vector. Looking at feature vectors independently and not combined, VH & VV was the best performing feature vector with Weather, Topology, and Elevation having significantly worse performance. VH & VV and Weather had much more features compared to Topology and Elevation, which should be taken into consideration.

	LGBM			FNN		
Features, Nearest	RMSE	Acc.	F1	RMSE	Acc.	F1
All	1.15	0.43	0.43	1.37	0.35	0.35
VH & VV	1.23	0.41	0.40	1.30	0.37	0.37
Weather	1.59	0.27	0.23	1.80	0.24	0.21
Topology	1.99	0.17	0.08	2.01	0.17	0.09
Elevation	1.91	0.20	0.14	1.98	0.17	0.10

Table 4.13: Summary of results for 2020 using harvest data with a resolution of 22×22 m^2, Nearest sampling, different feature vectors, and models. The best overall performance was achieved with LGBM using the All feature vector. Looking at feature vectors independently and not combined, VH & VV was the best performing feature vector with Weather, Topology, and Elevation having significantly worse performance. VH & VV and Weather had much more features compared to Topology and Elevation, which should be taken into consideration.

	LGBM			FNN		
Features, Nearest	RMSE	Acc.	F1	RMSE	Acc.	F1
All	1.34	0.38	0.38	1.72	0.32	0.32
VH & VV	1.46	0.36	0.35	1.75	0.32	0.32
Weather	1.93	0.23	0.20	2.33	0.19	0.17
Topology	2.88	0.18	0.10	2.91	0.19	0.11
Elevation	2.73	0.18	0.14	2.81	0.18	0.13

Chapter 5

Discussion

In the following chapter, we attempt to explain our results and discuss what we think possibly could have gone wrong at every step of the process. We also share some ideas of possible improvements for further research on this topic.

5.1 Data set

The 50×50 m^2 harvest data set is relatively small for a machine learning model. For each year we have ~ 5000 samples which in itself is a small number of samples for the training of a machine learning model, but even more so when considering that we have up to ~ 500 feature vectors when using all features. It might be difficult for our models to learn which of these features are important. However, if we compare the results from using the 50 m grid with higher resolutions (22 m and 12 m where Nearest sampling was used) we see a trend that decreases in performance. This is surprising, as the higher-resolution grids greatly increased the size of the data set (more than 4 times larger for 22 m compared to 50 m). These results would suggest that more new novel data is needed and that the size of the data set was not the issue.

A key difference between Sentinel-1 and Sentinel-2 is that there are only two bands (VH and VV) for Sentinel-1 and nine bands for Sentinel-2 [1]. These nine different bands correspond to imagery using different wavelengths, which may contain more information important for harvest prediction. This could explain the lower performance when using Sentinel-1 compared to using Sentinel-2, as done by authors in [1], where they achieved an average accuracy of 0.82. Our best accuracy score was 0.60, see Table 4.4. An interesting test that should be investigated in the future is how the combination of Sentinel-1 and Sentinel-2 performs. It may be possible to further improve the prediction score, even exceeding the performance when using only Sentinel-2.

5.1.1 Grid Sampling

The 3-by-3 Grid sampling introduces a lot of new features. For each coordinate, time point, and feature type (excluding Weather) we now have nine features instead of just one. This lowers the data set size per feature ratio significantly but without it, we are missing a lot of data relevant to each point. This could explain why we are not seeing an increase in performance when using Grid sampling compared to Nearest sampling for the 50 m grid. It may also suggest that the neighboring samples do not provide any new novel data to our models.

5.1.2 Weather, Elevation, & Topology

When we added weather data we expected to see a significant increase in performance since our intuition about agriculture says that weather is crucially important. This was not the case, as can be seen in Tables 4.3, 4.4, 4.5 and 4.6. Instead, we just saw a small but barely significant performance increase. This could be explained by the sparsity of the weather stations which can be seen in Fig. 3.15 which resulted in quite inaccurate weather data. A solution to deal with the sparsity could be to use a more sophisticated interpolation method such as Kriging interpolation, but this was outside of the scope of this thesis. Another explanation for the low impact of the weather data on the performance could be the data set size per feature ratio since it adds 5 new features per time step which accumulates to $16 \cdot 5$ new features for each sample.

Previous studies [33] have shown that altitude height data has some correlation with the yield. When we added the relatively high resolution (10 x 10 m^2) elevation data we also expected an increased performance, but again we saw no significant impact. The elevation data does not add many features to the data set. In fact, it only adds a single or nine features with 3-by-3 Grid sampling since we assume that elevation height does not change over time. The issue at play here could instead be data balancing since the proportion of elevation features are much smaller than the proportion of other features in the data.

Topology performed the worst of all features on its own. This could either mean that topology is not an important feature in harvest prediction or that our representation of topology classes as one-hot-encoding adds too many features. Again, this could be related to the data set size per feature ratio, making the model unable to learn the representation.

5.2 Despeckling

The SAR2SAR model which was used for despeckling the Sentinel-1 data is essentially a black box. It seems to be doing a good job of despeckling the data while maintaining sharpness. We saw that it improved our models' performance by about $5-15\%$ but this does not exclude the possibility of it removing valuable information. It is possible that there is a better way to despeckle Sentinel-1 data which could yield increased performance.

5.3 Model Selection

We did not invest a great amount of time trying to find new and potentially better models, which might have led to better results. The LGBM model was selected as a primary model throughout the project simply because of how GBDT models have shown previous good results in harvest prediction tasks with Sentinel-2. However, it is worth noting that GBDT models do not explicitly deal with spatial correlations in the same way as a Convolutional Neural Network (CNN) model such as a Temporal Convolutional Network (TCN) does. Spatial correlations are most likely very important to take into consideration in harvest prediction tasks. Since points that are close to each other in space will probably have similar harvests.

Changing the representation of the data set into a format usable for a TCN model is not a trivial task and is outside the scope of this thesis. It would involve grouping coordinates into small grids which could be interpreted as an image by the TCN. The model would then be able to learn the spatial correlation within that grid, and hopefully generalize these patterns to previously unseen data.

5.4 Results

A general pattern emerging from the results is that only Sentinel-1 data perform better on its own than the auxiliary weather and terrain features do. Still, we consistently get the best results when combining all features, although, not by much.

We were not able to produce a model with accuracy reaching the accuracy of models trained on Sentinel-2 data [1] with our 50×50 m^2 grid. However, we were still able to show that it might be feasible with a sufficiently large data set (See Fig. 4.1, 4.2, 4.3). Unfortunately, when we increased the data set size with the 22×22 m^2 grid we only saw a decrease in performance. We are not entirely sure why this happened. This is particularly strange when considering

the trends which suggested that the model was learning something from the 50 m data. Additionally, the same experiments for 12×12 m^2 are shown informally in the appendix A.3 and they show a further decrease in the performance. In a machine learning context, more data should lead to better results. We are unsure if this is due to an error by us or if there is something wrong with the higher-resolution harvest data. (Appendix A.4 have more results indicating that it is the higher-resolution harvest data at fault).

5.4.1 Distributions & Transfer Learning

We found it very interesting to compare the distributions of the true harvest and the distribution of the predicted harvest (see Figures 4.4, 4.5, 4.6, 4.7). The model seems to roughly capture the shape of the true distribution. This was an indication to us that the model was doing something reasonably intelligent and not just average guessing which would look more like a thin Gaussian.

However, we also see that the distributions vary significantly over the years. This variation has two possible explanations, the first and probably most significant is that our data set is too small and the number of farmers who have provided data to the data set varies between the years with an increasing trend. This makes it hard for us to model the true distribution for each year since we only have a glimpse of what a complete data set would look like. Secondly, In Jordbruksverkets database [21] we can see how the mean of a more complete data set should vary between each year. This data shows that the average harvest per area unit actually varies significantly over the years. A way of mitigating this problem, which should be investigated further, is to model the harvest distribution with some kind of algorithm. We have a few models in mind for this task, those being a: dynamical statistical model estimated with a Kalman filter or variational autoencoder. Being able to precisely model the harvest distribution would be greatly beneficial, and should be added to our models so that they can adapt to the variation.

This fact helps in explaining the poor results of the transfer learning on our data since we are training it on one distribution and testing it on a significantly different one. Ideally, this should not be a problem for a machine-learning model trained on a sufficient amount of data. But in reality, there are probably more hidden features that determine the distribution for a given year making it difficult to predict.

5.4.2 Year 2018

In 2018, Sweden had its warmest summer ever recorded [46]. It was also a dry summer with extremely low precipitation. The results of this extreme weather

event can be seen throughout our data and results. Notice how the harvest distribution for 2018 (see Fig. 4.5) differs significantly from the other years. The top features for 2018 from the Boruta algorithm included height in the top 5 features which were not seen in the other years and instead favored the raw Sentinel-1 data and sometimes weather.

Chapter 6

Conclusion

The primary conclusion of this thesis is that harvest prediction using Sentinel-1 radar backscatter is most likely feasible with a sufficiently large data set and an adequately powerful model. We have shown that Sentinel-1 data contains valuable information for harvest prediction. Further work is needed to investigate why lower-resolution harvest grid scores are better than those with the higher-resolution harvest grid.

The LGBM model outperformed the FNN model in all aspects. It consistently outperformed the FNN model in RMSE and classification scores by a small but yet significant amount. It was also several factors faster to train. We would also like to mention that we tried simple models such as linear regression and logistic regression, these performed worse than LGBM.

The indices had an interesting trend similar to the NDVI trend for Sentinel-2 in [1]. Adding indices to our models resulted in no increase in performance as was seen for Sentinel-2 [1].

The auxiliary data (Weather, Elevation, and Topology) gave a slight boost to performance. Weather and Elevation seem useful to a degree, as both showed up among the most important features. Topology, on the other hand, did not seem useful at all. It rarely showed up at all as a favored feature from the Boruta algorithm.

Transfer learning did not work very well at all. Our best-case scenario was when using the 2020 data set for training and predicting the data set from 2017. This still gave us an error of 1.69 tonnes per hectare which we consider insufficient for operational use. The poor performance in transfer learning is not surprising, as transfer learning (in the sense of extrapolating over time) is a difficult problem. The model would not only need to perform well for the distribution it was trained on, but it would also need to learn how the distribution changes in the future. Further work is needed to estimate how the harvest distribution varies between years to hopefully someday enable the operational use of these models.

6.1 Future works

Throughout this project, we came up with several ideas which we would like to try but realized that it was unfeasible to try them all within the time frame for the thesis. If anyone decides to continue our work then we would like to offer some advice on what we think is worth investigating:

Combine Sentinel-1 and Sentinel-2 data: The major next extension of our work is to combine Sentinel-1 data with Sentinel-2 data since it is possible that Sentinel-1's backscatter contains information that can not be derived from Sentinel-2's data and vice versa.

CNN model: The LGBM model does not take spatial correlations into account. Our intuition is that spatial correlations are most likely very important in harvest prediction. Further work should investigate the possibility of using a CNN-based model, which can model spatial correlations, and see how it compares with the LGBM model.

Transfer learning: There should be a more thorough investigation of why the transfer learning did not work and how it can be improved. Within the project, we discussed if it is possible to predict the next year's distribution given previous years' distribution together with any auxiliary data. Furthermore, it should be investigated if a predicted distribution can be used to predict the harvest at specific points.

Other crops: We believe that this approach of coordinate-wise harvest prediction with Sentinel-1 radar data could be expanded to other crops. Winter wheat was picked due to the possibility of comparing previous harvest prediction results using Sentinel-2 optical imagery [1].

RFI-filter: The RFI-filter we constructed is very primitive. It is worth investigating if it is possible to either improve it, such that it would remove more noise and less good data. It is also worth investigating if it is possible for an ML model to both remove the noise and possibly reconstruct the missing data. This could open up the possibility of using a finer time interval with 2 or 3 images per week.

Appendix A

Latest Findings

In this part of the report, we will present our latest results which we could not formally include in the report, due to time limitations.

A.1 Results in a Compact Form

Let $\Psi \in \{\text{G-50}, \text{N-50}, \text{N-22}, \text{N-12}\}$ where G = Grid, N = Nearest and 50, 22, and 12 is the harvest grid resolution in meters and $\bar{\mathbf{x}}$ = Feature Vector.

Table A.1: RMSE for each year, model, feature vector, sampling method, and resolution. Note that the LGBM model outperforms the FNN model in all cases.

		2017		2018		2019		2020	
Ψ	$\bar{\mathbf{x}}$	LGBM	FNN	LGBM	FNN	LGBM	FNN	LGBM	FNN
G-50	All	0.80	1.09	0.75	0.87	0.93	1.07	1.08	1.37
	VH-VV	0.85	1.12	0.81	0.87	1.01	1.15	1.20	1.47
N-50	All	0.81	1.05	0.76	0.87	0.93	1.13	1.09	1.37
	VH-VV	0.86	1.12	0.81	0.88	1.04	1.12	1.25	1.53
N-22	All	1.07	1.26	0.89	1.06	1.15	1.37	1.34	1.72
	VH-VV	1.11	1.25	0.93	1.02	1.23	1.30	1.46	1.75
N-12	All	1.31	1.49	1.00	1.14	1.37	1.50	1.53	1.81
	VH-VV	1.34	1.53	1.02	1.11	1.42	1.52	1.63	1.92

A.2 Harvest Distribution 22 m vs. 50 m

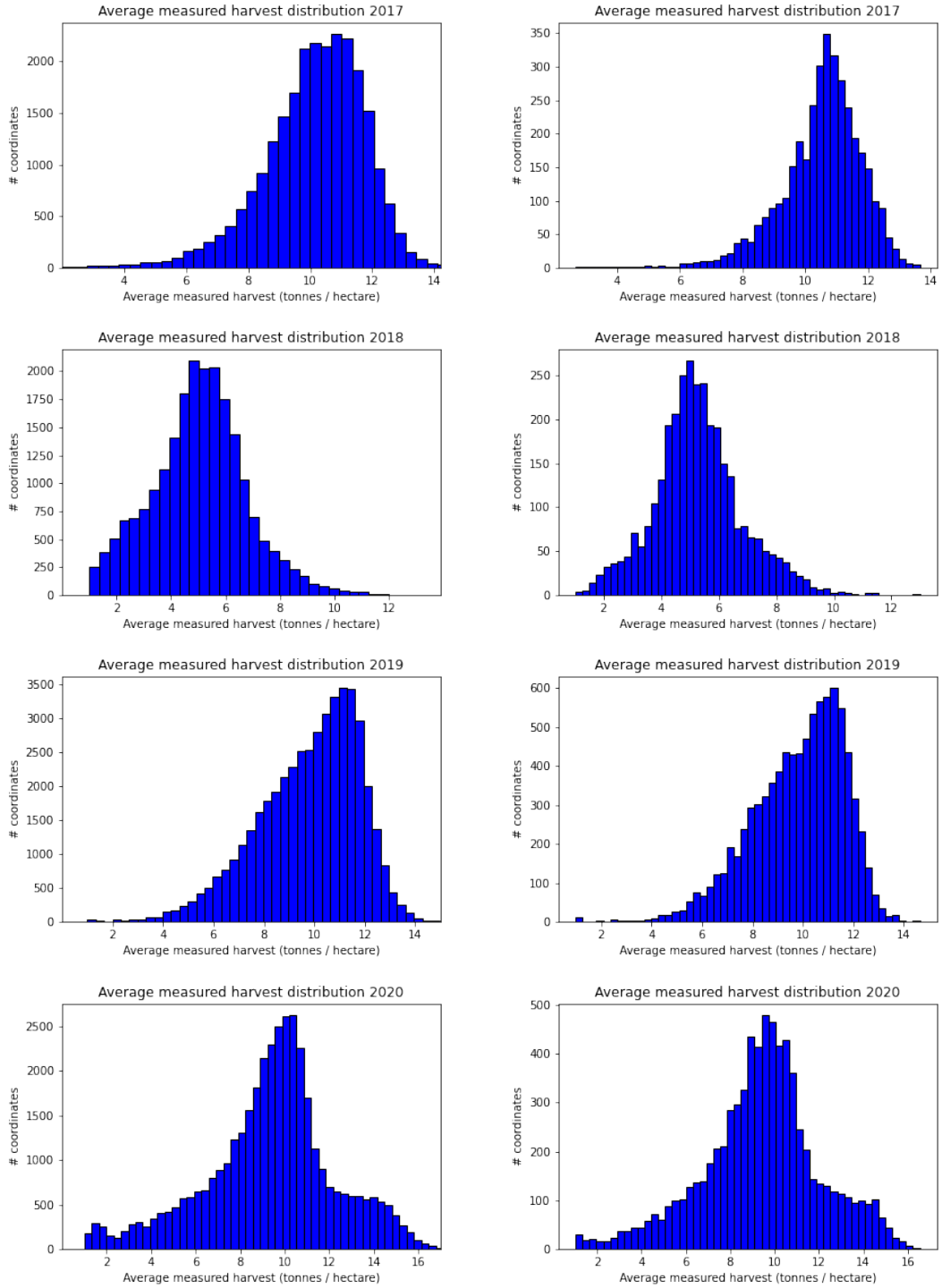

Figure A.1: True harvest distributions for 22 m grid (left) compared with 50 m (right) grid

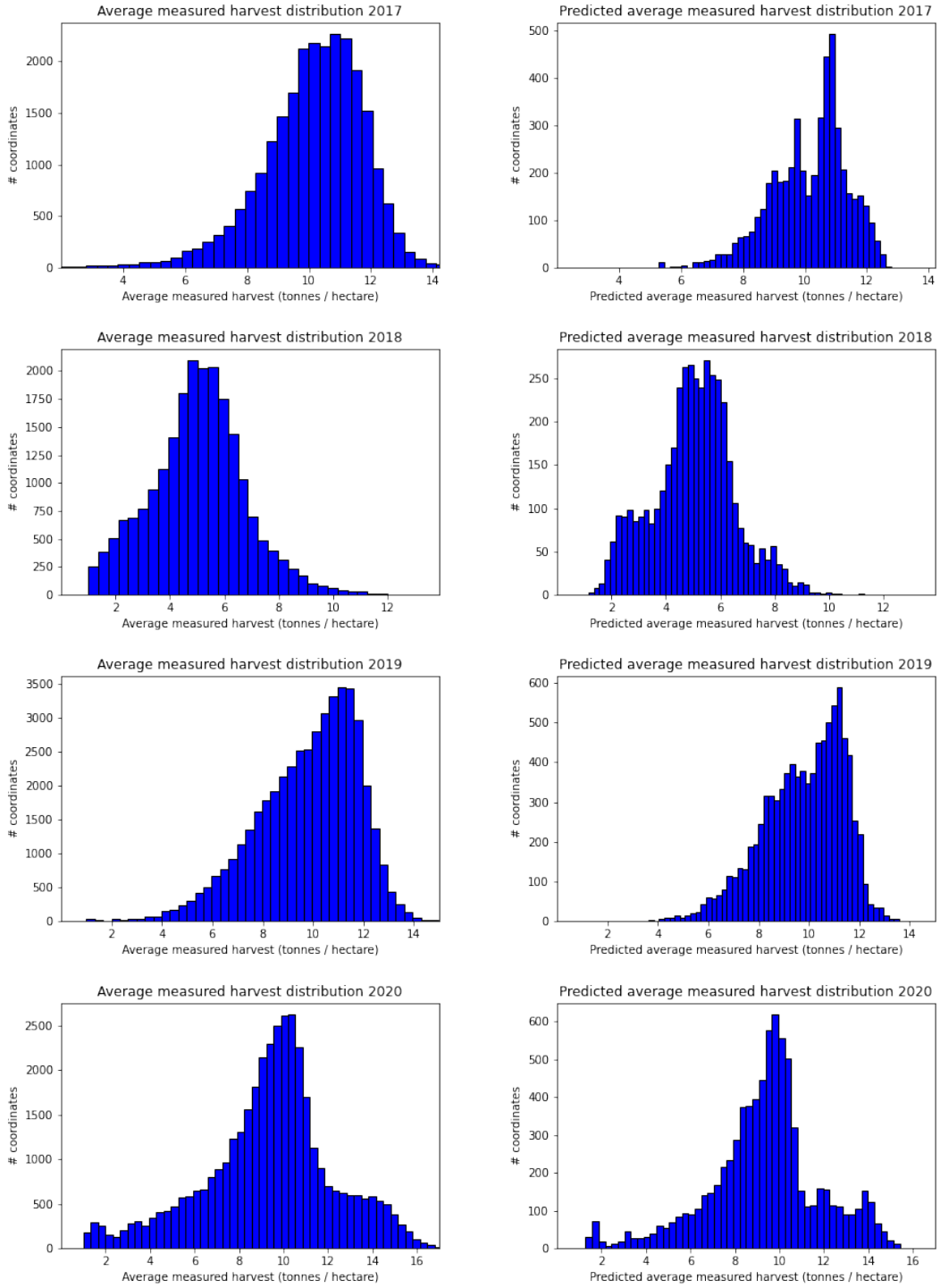

Figure A.2: True harvest distributions for 22 m grid compared with the predicted harvest distributions

A.3 General Results (12 m Resolution)

Table A.2: Summary of results for 2017 using harvest data with a resolution of 12×12 m^2, Nearest sampling, different feature vectors, and models.

	LGBM			FNN		
Features, Nearest	RMSE	Acc.	F1	RMSE	Acc.	F1
All	1.31	0.43	0.41	1.49	0.37	0.36
VH & VV	1.34	0.42	0.40	1.53	0.36	0.34
Weather	1.56	0.35	0.32	1.60	0.33	0.29
Topology	1.84	0.25	0.16	1.86	0.26	0.17
Elevation	1.72	0.30	0.25	1.80	0.27	0.19

Table A.3: Summary of results for 2018 using harvest data with a resolution of 12×12 m^2, Nearest sampling, different feature vectors, and models.

	LGBM			FNN		
Features, Nearest	RMSE	Acc.	F1	RMSE	Acc.	F1
All	1.00	0.50	0.49	1.14	0.42	0.41
VH & VV	1.02	0.49	0.48	1.11	0.45	0.44
Weather	1.43	0.32	0.27	1.47	0.31	0.24
Topology	1.74	0.26	0.17	1.76	0.26	0.18
Elevation	1.57	0.30	0.24	1.64	0.28	0.22

Table A.4: Summary of results for 2019 using harvest data with a resolution of 12×12 m^2, Nearest sampling, different feature vectors, and models.

	LGBM			FNN		
Features, Nearest	RMSE	Acc.	F1	RMSE	Acc.	F1
All	1.37	0.39	0.38	1.50	0.35	0.34
VH & VV	1.42	0.38	0.36	1.52	0.33	0.32
Weather	1.75	0.26	0.23	1.86	0.24	0.21
Topology	2.15	0.17	0.08	2.17	0.16	0.08
Elevation	2.05	0.19	0.14	2.09	0.17	0.11

Table A.5: Summary of results for 2020 using harvest data with a resolution of 12×12 m^2, Nearest sampling, different feature vectors, and models.

	LGBM			FNN		
Features, Nearest	RMSE	Acc.	F1	RMSE	Acc.	F1
All	1.53	0.36	0.36	1.81	0.31	0.30
VH & VV	1.63	0.34	0.33	1.92	0.31	0.30
Weather	2.10	0.22	0.20	2.45	0.21	0.18
Topology	3.06	0.17	0.10	3.08	0.18	0.10
Elevation	2.87	0.17	0.13	3.01	0.16	0.12

A.4 Erroneous Code or Harvest Data?

As mentioned in the thesis, we would expect that the error would be lower or at least the same for the higher resolutions grid (N-22 and N-12). Hence, there may be either a problem with the code or the finer resolution harvest data. We suspect that the code working correctly and that the higher-resolution harvest data is the problem. This is due to several reasons. If we have an error in the code, then that would reflect on the 50 m grid performance, which it does not. Further, we tested this empirically by creating two new data sets using randomness instead of the actual satellite data for the input data and the 50 m harvest data as labels. The randomness was done in two ways, creating two new feature vectors for N-50. The first feature vector we called Random, $\bar{\mathbf{x}} = \mathrm{R}$, and it replaced the correct VH and VV values with samples from a uniform distribution, $\mathbf{U}(0, 1)$. The second feature vector we called Patch Random, $\bar{\mathbf{x}} = \mathrm{PR}$, and it replaced the correct VH and VV values with random VH and VV values from the same patch. This randomness approach tested what would happen if the data was from Sentinel-1 satellites, but geo-referenced incorrectly. This means that the satellite data was not completely random, and was limited to a similar area of the Earth.

Our thought was that the Random feature vector would just guess the average, as there would be no correlation between the input VH and VV values and the labels. This was confirmed and can be seen in Fig. A.4.

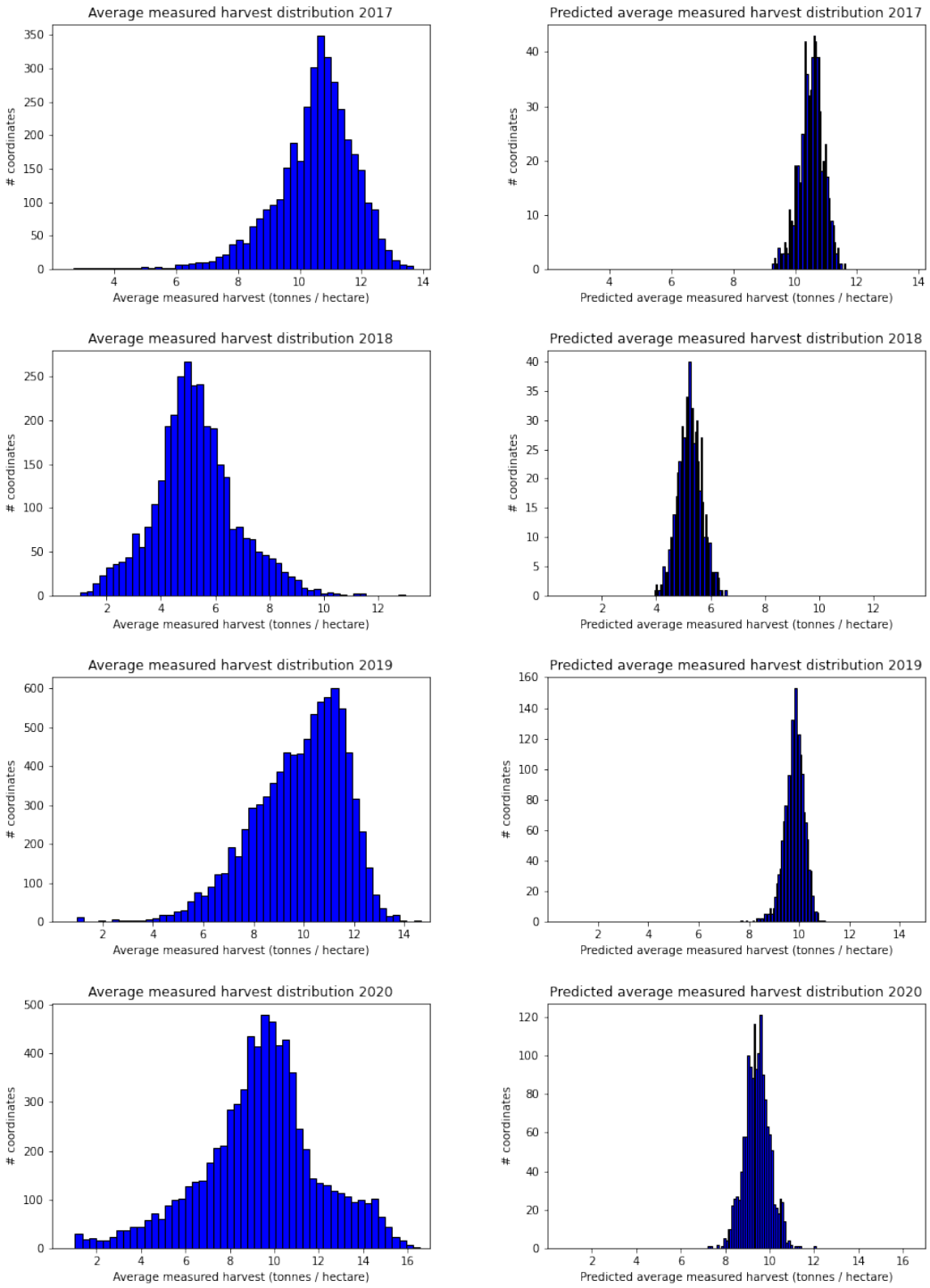

Figure A.3: True harvest distributions for 50 m grid (left) compared with predicted distribution (right) using Ψ = N-50 and $\bar{\mathbf{x}}$ = R.

For the Patch Random feature vector, our thought was that getting random samples from within the same patch could potentially lead to a more complex predicted distribution. This could be seen for 2020, but otherwise, the predicted distribution looked more like average guesses as seen when it was completely random Fig. A.4. This is further evidence that the code was not at fault.

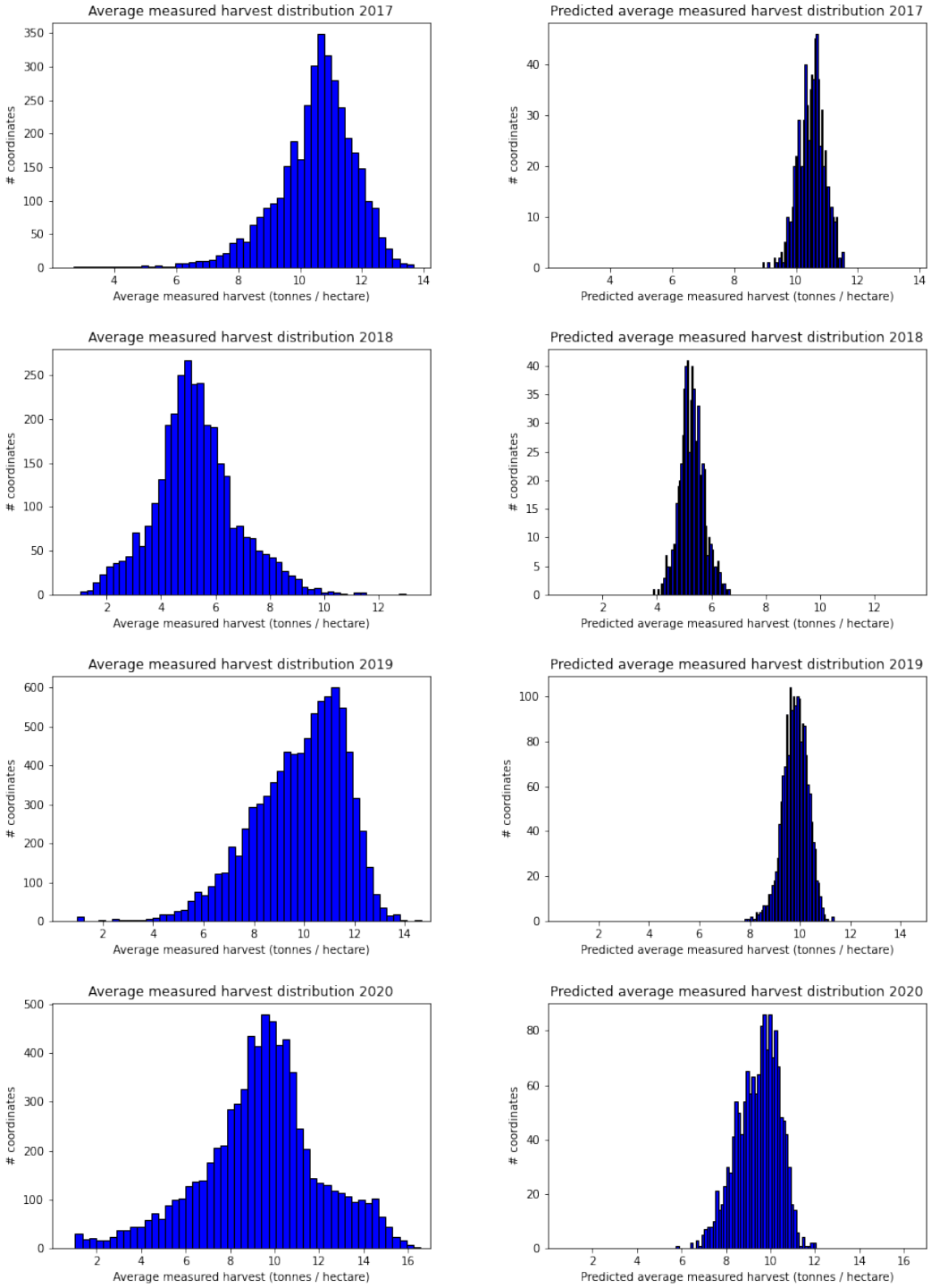

Figure A.4: True harvest distributions for 50 m grid (left) compared with predicted distribution (right) using Ψ = N-50 and $\bar{\mathbf{x}}$ = PR.

The table below shows the results from the different random feature vectors using the LGBM model and N-50, see Table A.6. Guessing the average harvest label was not a bad choice, as the RMSE was not that high. However, we see a large decrease in the classification metrics, as these are more sensitive to the variance of the underlying distribution.

Table A.6: The results using the LGBM model and two different random feature vectors for the input data. To get the classification metrics an error margin of 0.5 tonnes per hectare was used.

		2017			2018		
Ψ	$\bar{\mathbf{x}}$	RMSE	Acc.	F1	RMSE	Acc.	F1
N-50	R	1.36	0.31	0.25	1.60	0.27	0.21
N-50	PR	1.37	0.32	0.26	1.60	0.28	0.22
		2019			2020		
Ψ	$\bar{\mathbf{x}}$	RMSE	Acc.	F1	RMSE	Acc.	F1
N-50	R	1.85	0.19	0.12	2.73	0.18	0.12
N-50	PR	1.82	0.19	0.13	2.60	0.18	0.14

During the thesis, we mostly worked with the 50 m harvest data as the finer resolution grids were hard to create due to the great number of data points. We did have an error in our sampling for 50 m, which we fixed. The issue was that the latitude coordinate was flipped when we geo-referenced it with the Sentinel-1 data, leading to the longitude being correct but not the latitude. Once this bug was fixed, we repeated the experiments. The performance increased by about 10 percent units. We were surprised that the error did not improve more, as this was not a small bug. However, this further showed the spatial correlation aspect of the problem and why a CNN-based model would perform well.

A.5 Can We Overfit?

At some point in the project, we wanted to see if it was possible to overfit the harvest data, not for any particular reason or with a particular goal in mind. But we thought that it might be interesting to include these results as well in the appendix, we clearly see that the model can overfit in the figures below. However, the overfitting does not decrease the validation error by a significant amount, as expected. This shows that the use of optimization techniques, such as drop-outs, could further minimize the generalization gap.

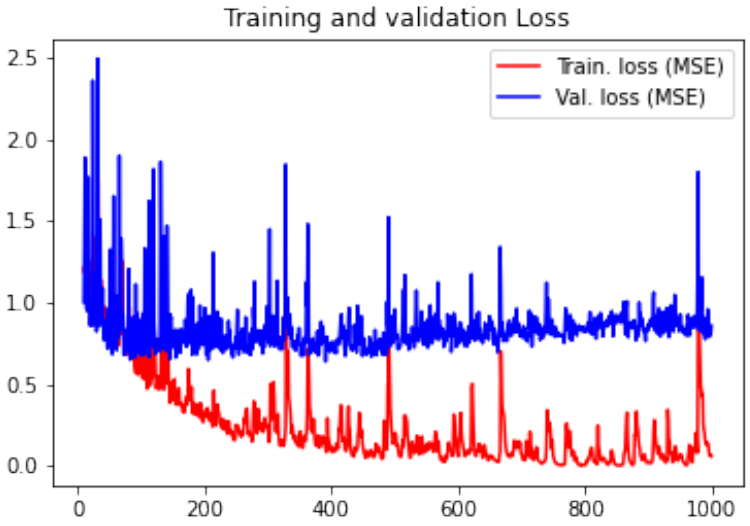

Figure A.5: Using 22 m grid and the FNN described below.

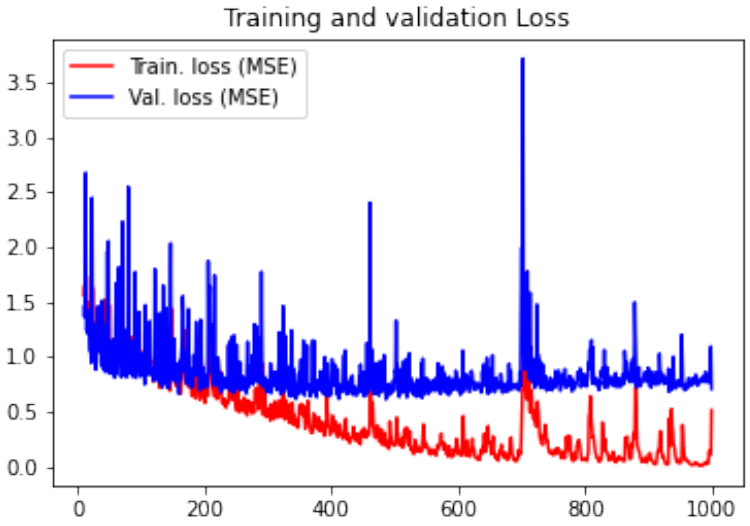

Figure A.6: Using 50 m grid and the FNN described below.

Here is the extended FNN used to achieve the overfitting:

```
n_features = X_train.shape[1]
inputs = keras.Input(shape=(n_features,))
x = layers.Dense(max(n_features*11, 500), activation="relu")(
                                    inputs)
x = layers.Dense(max(n_features*9, 500), activation="relu")(x)
x = layers.Dense(max(n_features*9, 500), activation="relu")(x)
x = layers.Dense(max(n_features*9, 500), activation="relu")(x)
x = layers.Dense(max(n_features*9, 500), activation="relu")(x)
x = layers.Dense(max(n_features*9, 500), activation="relu")(x)
```

```
x = layers.Dense(max(n_features*9, 500), activation="relu")(x)
x = layers.Dense(max(n_features*9, 500), activation="relu")(x)
x = layers.Dense(max(n_features*9, 500), activation="relu")(x)
x = layers.Dense(max(n_features*9, 500), activation="relu")(x)
x = layers.Dense(max(n_features*8, 500), activation="relu")(x)
x = layers.Dense(max(n_features*7, 500), activation="relu")(x)
x = layers.Dense(max(n_features*6, 500), activation="relu")(x)
x = layers.Dense(max(n_features*5, 500), activation="relu")(x)
x = layers.Dense(max(n_features*5, 500), activation="relu")(x)
x = layers.Dense(max(n_features*5, 500), activation="relu")(x)
x = layers.Dense(max(n_features*4, 500), activation="relu")(x)
x = layers.Dense(max(n_features*3, 300), activation="relu")(x)
x = layers.Dense(max(n_features*2, 200), activation="relu")(x)
# Max 200, min 50 nodes
x = layers.Dense(min(max(int(n_features* 1/2), 50), 200),
                                    activation="relu")(x)
# Max 50, min 20 nodes
x = layers.Dense(min(max(int(n_features* 1/5), 20), 50),
                                    activation="relu")(x)
outputs = layers.Dense(1)(x)

model = keras.Model(inputs=inputs, outputs=outputs, name="
                                    MyFNN_regressor")

model.compile(
    optimizer=keras.optimizers.Adam(),
    loss=keras.losses.MeanSquaredError(),
)

model.summary()


epochs = 100*10
batch_size = 248
history = model.fit(
    X_train,
    y_train,
    batch_size=batch_size,
    validation_split=0.1,
    epochs=epochs,
    verbose=2,
    #callbacks=callbacks,
)
```

Appendix B

Miscellaneous

B.1 Sentinel-1 Download Configuration

We used the `eo-learn` python package [9] for access to Sentinel 1 data. Using our coordinates from the field data we could locate which EO patches covered the fields. Using the `SentinelHubInputTask` class, we downloaded the data with the following parameters:

```
processing_params = {
    "backCoeff": "GAMMA0_TERRAIN",
    "orthorectify": True,
    "demInstance": "COPERNICUS",
    "downsampling": "BILINEAR",
    "upsampling": "BILINEAR"
}

data_task = SentinelHubInputTask(
    data_collection=DataCollection.SENTINEL1_IW,
    bands_feature=(FeatureType.DATA, "IW"),
    bands=["VH", "VV"],
    config=config,
    time_difference=datetime.timedelta(minutes=120),
    max_threads=4,
    resolution=11.0,
    aux_request_args = {'processing': processing_params}
)
```

The variable `processing_params` contains parameters for orthorectification and radiometric calibration [42].

B.2 Calculating Class-Metrics from RMSE

```
import numpy as np
from sklearn import metrics
import numpy.typing as npt

def class_metrics(y_test, y_hat, err_tol=0.5):
    y_test = np.array(y_test)
    y_hat = np.array(y_hat)

    y_test_class = y_test.round().astype(int)
    y_hat_class = np.zeros(len(y_hat)) - 1

    for i in range(len(y_test)):
        y_t = 6.51
        y_h = 7.1

        if np.abs(y_test[i] - y_hat[i]) <= err_tol:
            y_hat_class[i] = y_test_class[i]

        else:
            y_hat_class[i] = int(np.round(y_hat[i]))

            if y_hat_class[i] == y_test_class[i]:
                if y_test[i] < y_hat[i]:
                    y_hat_class[i] = y_hat_class[i] + 1
                else:
                    y_hat_class[i] = y_hat_class[i] - 1

    acc = metrics.accuracy_score(y_test_class, y_hat_class)
    f1 =  metrics.f1_score(y_test_class, y_hat_class, average='
                                    weighted')
    return acc, f1
```

B.3 Authors' Contact Information

Oliver Persson Bogdanovski:

- oliver.persson.b@gmail.com
- +46763169956

Christoffer Svenningsson:

- christoffer.c.svenningsson@gmail.com

Bibliography

[1] Camilla Broms, Mikael Nilsson, Andreas Oxenstierna, Alexandros Sopasakis, and Karl Åström. Combined analysis of satellite and ground data for winter wheat yield forecasting. *Smart Agricultural Technology*, 3:100107, 2023.

[2] Mike Daily, Swarup Medasani, Reinhold Behringer, and Mohan Trivedi. Self-driving cars. *Computer*, 50(12):18–23, 2017.

[3] Emanuele Dalsasso, Loic Denis, and Florence Tupin. SAR2sar: A semi-supervised despeckling algorithm for SAR images. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 14:4321–4329, 2021.

[4] Daniel Homola. Python implementation of boruta feature selection. `https://pypi.org/project/Boruta/`. Accessed: 2022-11-30.

[5] DeepMind Technologies Limited. Alphafold can accurately predict 3d models of protein structures and is accelerating research in nearly every field of biology. `https://www.deepmind.com/research/highlighted-research/alphafold`. Accessed: 2022-08-22.

[6] Li Deng. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

[7] Niklas Donges. Transfer learning. `https://builtin.com/data-science/transfer-learning`, 2022. Accessed: 2022-11-10.

[8] L. Denis E. Dalsasso and F. Tupin. Sar2sar gitlab. `https://gitlab.telecom-paris.fr/ring/sar2sar`, 2021. Accessed: 2022-10-11.

[9] EO Research team. eo-learn. `https://eo-learn.readthedocs.io/en/latest/`. Accessed: 2022-10-12.

[10] Fernando Pérez, Brian Granger, et al. jupyter: Free software, open standards, and web services for interactive computing across all programming languages. `https://jupyter.org/`. Accessed: 2022-11-28.

[11] François Chollet et al. Keras: Deep learning for humans. `https://keras.io/`. Accessed: 2022-11-28.

[12] Gary Sherman et al. Qgis: A free and open source geographic information system. `https://www.qgis.org/en/site/`. Accessed: 2022-11-28.

[13] Ian J. Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, Cambridge, MA, USA, 2016. `http://www.deeplearningbook.org`.

[14] Google LLC. How our business works. `https://about.google/intl/en_SE/how-our-business-works/`. Accessed: 2022-11-18.

[15] Google LLC. Tensorflow: An end-to-end machine learning platform. `https://www.tensorflow.org/`. Accessed: 2022-11-28.

[16] Guido van Rossum et al. The python programming language. `https://www.python.org/`. Accessed: 2022-11-28.

[17] Dave Hudak, Doug Johnson, Alan Chalker, Jeremy Nicklas, Eric Franz, Trey Dockendorf, and Brian L McMichael. Open ondemand: a web-based client portal for hpc centers. *Journal of Open Source Software*, 3(25):622, 2018.

[18] Mohammad Rafiqul Islam and Nguyet Nguyen. Comparison of financial models for stock price prediction. *Journal of Risk and Financial Management*, 13(8):181, 2020.

[19] Jarek Jasiewicz and Tomek Stepinski. Geomorphon. `https://grass.osgeo.org/grass82/manuals/r.geomorphon.html`, 2022. Accessed: 2022-09-01.

[20] John D. Hunter et al. Matplotlib: Visualization with python. `https://matplotlib.org/`. Accessed: 2022-11-28.

[21] Jordbruksverket. Hektar- och totalskörd efter län och gröda. `https://statistik.sjv.se/PXWeb/pxweb/sv/Jordbruksverkets%20statistikdatabas/Jordbruksverkets%20statistikdatabas__Skordar/JO0601J01.px/`. Accessed: 2022-11-21.

[22] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[23] Keras. Adam optimizer documentation. `https://keras.io/api/optimizers/adam/`. Accessed: 2022-11-09.

[24] Saeed Khabbazan, Paul Vermunt, Susan Steele-Dunne, Lexy Ratering Arntz, Caterina Marinetti, Dirk van der Valk, Lorenzo Iannini, Ramses Molijn, Kees Westerdijk, and Corné van der Sande. Crop monitoring using sentinel-1 data: A case study from the netherlands. *Remote Sensing*, 11(16), 2019.

[25] Konstantina Kourou, Themis P Exarchos, Konstantinos P Exarchos, Michalis V Karamouzis, and Dimitrios I Fotiadis. Machine learning applications in cancer prognosis and prediction. *Computational and structural biotechnology journal*, 13:8–17, 2015.

[26] Miron B. Kursa and Witold R. Rudnicki. Feature selection with the boruta package. *Journal of Statistical Software*, 36(11):1–13, 2010.

[27] Lantmännen. Precision agriculture. `https://www.lantmannen.com/research-and-innovation/innovation-from-field-to-fork/innovative-cultivation-methods/precision-agriculture/`, 2022. Accessed: 2022-09-02.

[28] Jaakko Lehtinen, Jacob Munkberg, Jon Hasselgren, Samuli Laine, Tero Karras, Miika Aittala, and Timo Aila. Noise2noise: Learning image restoration without clean data, 2018.

[29] Xiangguang Leng, Kefeng Ji, and Gangyao Kuang. Radio frequency interference detection and localization in sentinel-1 images. *IEEE Transactions on Geoscience and Remote Sensing*, 59(11):9270–9281, 2021.

[30] Shohreh Liaghat and Siva Balasundram. A review: The role of remote sensing in precision agriculture. *American Journal of Agricultural and Biological Science*, 5, 01 2010.

[31] Microsoft Corporation. Light gradient boosting machine. `https://github.com/microsoft/LightGBM`. Accessed: 2022-11-28.

[32] Microsoft Corporation. Lightgbm regressor, python api. `https://lightgbm.readthedocs.io/en/latest/pythonapi/lightgbm.LGBMRegressor.html`. Accessed: 2022-11-10.

[33] Thomas Minda, Michiel Van der Molen, Paul Struik, M. Combe, Pedro Jimenez, Muhammad Khan, and Jordi Arellano. The combined effect of elevation and meteorology on potato crop dynamics: A 10-year study in the gamo highlands, ethiopia. *Agricultural and Forest Meteorology*, 262:166–177, 11 2018.

[34] Tom M Mitchell. *Machine learning*, volume 1. McGraw-hill New York, 1997.

[35] MZ Naser and Amir Alavi. Insights into performance fitness and error metrics for machine learning. *arXiv preprint arXiv:2006.00887*, 2020.

[36] Ali Nasrallah, Nicolas Baghdadi, Mohammad El Hajj, Talal Darwish, Hatem Belhouchette, Ghaleb Faour, Salem Darwich, and Mario Mhawej. Sentinel-1 data for winter wheat phenology monitoring and mapping. *Remote Sensing*, 11(19):2228, 2019.

[37] Chalmers University of technology. Alvis. `https://www.c3se.chalmers.se/about/Alvis/`, 2022. Accessed: 2022-11-04.

[38] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[39] SciKit-Learn. Cross-validation: evaluating estimator performance. `https://scikit-learn.org/stable/modules/cross_validation.html`. Accessed: 2022-10-20.

[40] SciKit-Learn. Metrics and scoring: quantifying the quality of predictions. `https://scikit-learn.org/stable/modules/model_evaluation.html#`. Accessed: 2022-11-08.

[41] scikit-learn. A random forest classifier. `https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html`. Accessed: 2022-11-30.

[42] Sentinel Hub. Sentinel-1 grd. `https://docs.sentinel-hub.com/api/latest/data/sentinel-1-grd/`. Accessed: 2022-10-11.

[43] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[44] Skogsstyrelsen. Skogliga grunddata. `https://www.skogsstyrelsen.se/sjalvservice/karttjanster/skogliga-grunddata/`, 2022. Accessed: 2022-09-01.

[45] Merrill Ivan Skolnik. *Radar Handbook*, pages 1.14, 1.17. McGraw-Hill Professional, 1990.

[46] SMHI. Sommaren 2018 - extremt varm och solig. `https://www.smhi.se/klimat/klimatet-da-och-nu/arets-vader/sommaren-2018-extremt-varm-och-solig-1.138134`, 2018. Accessed: 2022-11-16.

[47] South Dakota State University. Radiometric calibration. `https://www.sdstate.edu/image-processing-lab/radiometric-calibration`. Accessed: 2022-10-12.

[48] Springer. Precision ag definition. `https://www.springer.com/journal/11119/updates/17240272`, 2019. Accessed: 2022-09-02.

[49] Christoffer Svenningsson and Oliver Persson. b. Github repository for thesis. `https://github.com/Christoffer-Svenningsson/sentinel_1_harvest_prediction`. Accessed: 2022-12-08.

[50] The European Space Agency. Sentinel-1. `https://sentinel.esa.int/web/sentinel/missions/sentinel-1`. Accessed: 2022-10-11.

[51] The New York Times. An invisible cage: How china is policing the future. `https://www.nytimes.com/2022/06/25/technology/china-surveillance-police.html`. Accessed: 2022-10-14.

[52] Travis Oliphant et al. Numpy: The fundamental package for scientific computing with python. `https://numpy.org/`. Accessed: 2022-11-28.

[53] Wes McKinney et al. pandas: Python data analysis library. `https://pandas.pydata.org/`. Accessed: 2022-11-28.

[54] Tzu-Tsung Wong. Performance evaluation of classification algorithms by k-fold and leave-one-out cross validation. *Pattern Recognition*, 48(9):2839–2846, 2015.

[55] Brett Wujek, Patrick Hall, and Funda Günes. Best practices for machine learning applications. *SAS Institute Inc*, 2016.

[56] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization (2016). *arXiv preprint arXiv:1611.03530*, 2017.

Master's Theses in Mathematical Sciences 2022:E74
ISSN 1404-6342

LUTFMA-3491-2022

Mathematics
Centre for Mathematical Sciences
Lund University
Box 118, SE-221 00 Lund, Sweden

http://www.maths.lth.se/