Universiteit Leiden
The Netherlands

Opleiding Informatica

Using machine learning to predict rainwater damage in urban areas

Christie Bavelaar

Supervisors:
Jan N. van Rijn, Mitra Baratchi

External Supervisor:
Ton Beenen (Stichting RIONED)

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
www.liacs.leidenuniv.nl

23/06/2021

Abstract

This thesis provides an application of machine learning techniques to a novel domain. First, we use multiple data sources to create a data set. We have used both Twitter message and alerts from the p2000 emergency network as target attribute to predict rainwater damage. Data sets from the "Actueel Hoogtebestand Nederland" and the Royal Dutch Meteorological Institute (KNMI) are used to engineer relevant features. We have explored three different sampling methods to provide an equal number of negative examples to the positive examples from the target data set. This way we train our model on a balanced data set. We then use machine learning to predict events of rainwater damage. We have measured the performance increase when including attributes on terrain height. We have performed experiments on different balanced data sets and different feature subsets. The highest mean accuracy we have achieved during our experiments was 77%. The sampling method which we find best supports the classification task achieved a mean accuracy of 58%. We have found a statistically significant improvement when adding terrain height attributes as features.

Contents

1 Introduction & Background

The Netherlands likes to view itself as a nation battling the elements. This fight has mainly been focused on preventing flooding from the rivers or the sea. When we focus our attention on just building the most advanced dikes we overlook the water coming from the sky.

1.1 Background

When a building's interior is damaged by rain it can be by water leaking down through the roof, or flooding from the street. Rain damage does not limit itself to areas near the coast or rivers, it can occur in any place experiencing heavy rainfall. Lower situated areas may have a slightly increased risk, because water accumulates there. These areas will also have a stormwater drainage system relying on pumps to drain the water, creating an extra point the system could fail. This is not to say that the capacity of a stormwater drainage system without pumps cannot be exceeded. Extreme rainfall can result in pluvial flooding. Pluvial flooding is a particularly tricky problem as it does not occur very often, but when it does it causes a great amount of damage. The city experiences direct damage to infrastructure, but also indirect damage from traffic congestion and health hazard when water flows from the drainage system back onto the street. Estimates aiming to quantify this damage differ, but are in the millions with the pluvial flood in Dortmund, Germany, 2008 leading to (EUR) 17.2 million in damage. In Hersbruck, Germany, in 2005, this was (EUR) 2.8 million [Rözer et al., 2016]. This extend of damage highlights the importance of predicting these events. If the risk of pluvial flooding is identified in time the city and its inhabitants can take preventive measures to reduce the damage of this heavy rainfall event. Since pluvial floods can occur anywhere, but do not occur very often it is difficult to identify risks in advance. Warning systems for flooding caused by rivers or the sea are much further developed. When a pluvial flood is detected in time traditional preventive flood measures are of little use. One of these methods would be to use sandbags to prevent the water from entering buildings. In the case of heavy rainfall, however, water enters a building mainly through the walls, roof, or toilets [Dekker et al., 2016]. Measures that do reduce the damage in case of heavy rainfall are a disconnection of stormwater and sewer networks [Sušnik et al., 2015], moving furniture to different floors [Dekker et al., 2016] or installing a backflow protection device [Rözer et al., 2016]. The usefulness of preventive and emergency measures depends on the city's infrastructure and the awareness of the population. In case of the floods in Hull, United Kingdom, 2008 there was no system of warning from surface water flooding in the United Kingdom [Coulthard and Frostick, 2010]. The questionnaire conducted by Dekker et al. [2016] after the extreme rainfall in Amsterdam in 2014 showed that 60% of respondents were not aware of the pending heavy rain [Dekker et al., 2016]. These examples show a need for more awareness and preventive action to reduce damage from heavy rainfall.

1.2 Research question

We can usually predict a heavy rainstorm a few hours in advance, but we have little insights into which level of rainfall and which terrain characteristics leave an area vulnerable to rainwater damage. Most research in this domain uses hydraulic models to predict the flow of rainwater, machine

learning techniques remain relatively unexplored. This thesis provides an application of machine learning techniques to a novel domain. We have used multiple open-source data sets to engineer our target and features. Most events of rainwater damage occur in urban areas, so this is where our research will focus on. Our research question is: How can machine learning techniques, combined with data from multiple sources be used to predict damage from rainfall in urban areas? We will do this by looking at several subquestions:

- Which data set is most reliable to use as a target attribute?
- Which sampling method best supports the classification task?
- Does adding terrain height attributes improve the performance of the model?

1.3 Contributions

We do not have perfect knowledge of all past instances of rainwater damage in the Netherlands. So we need a data set to use as a target attribute that will approximate reality as well as possible. We have compared two different data sets to see which would be most useful for our classification task. The first data set contains twitter damages referencing rainwater damage and the second data set contains emergency alerts of water damage. The target data set only describes positive cases of rainwater damage. So, we needed some method to gather negative examples as well. We have compared three different sampling methods to find out which one best supports our classification task. The first sampling method is fully random, the second is based on Dutch addresses and the final sampling method uses equal locations for positive and negative examples. We compare the data sets resulting from these sampling methods and the performance of the models trained on them to find which method best supports our classification task. We have incorporated terrain height features by taking height measurements for a 100 x 100m area around an example as features. Experiments are conducted to find out if including these features results in a better performance. We have used rainfall measured on a daily basis for this research. We have also explored the possibilities of adding features expressing the rainfall per hour. Unfortunately, the assumptions made earlier in the project caused the framework to be ill-suited to this level of detail. An overview of our approach, as well as an explanation of the results, can be found in the appendix A

1.4 Thesis overview

We first discuss academic work in the domain of pluvial flooding and machine learning related to this thesis is discussed in section 2. We describe the data sets, pre-processing steps, and machine learning methods we have used in the Data ensembling, pre-processing & modeling, section 3. The experiments we have performed are outlined in section 4, their results are described in section 5. These results are discussed and explained in Discussion and Limitations, 6. Section 7 provides our final conclusions and recommendations for further research.

2 Related Work

We will first review work on previous events of pluvial flooding. We then look at hydraulic models predicting rainwater flow. We continue with a review of various machine learning methods and end with a review of previous work combining the prediction of rainwater damage with machine learning techniques.

2.1 Pluvial flooding

Previous research focuses heavily on individual cases. [Dekker et al., 2016, Spekkers et al., 2017, Rözer et al., 2016]. These authors use various forms of questionnaires to determine the extent of the damage and preparedness of the population. These researchers find that most households are not aware of the severity of the upcoming rainfall and the risks they are exposed to. The damage caused by the extreme rainfall events is difficult to compare. Rözer et al. [2016] find that damage mostly occurred in buildings with basements during the flooding of Hersbruck and Lohmar in 2005 and Osnabruck in 2010. After analysing the damage from heavy rainfall in Amsterdam in 2014 Dekker et al. [2016] conclude that while damage in basements is the greatest, the most instances of damage are to walls and roofs.[Dekker et al., 2016]. This difference may be due to factors unique to the city experiencing heavy rainfall. Spekkers et al. [2017] compared this same event in Amsterdam to another case of extreme rainfall in Munster, Germany also in 2014. They found the same difference with walls and roofs being most affected in Amsterdam where basements were in Munster [Spekkers et al., 2017]. Coulthard and Frostick [2010] take a slightly different approach by focusing not on the preventive measures taken by households, but on the urban drainage system during a case of pluvial flood in Hull, United Kingdom in 2008. They found that lower-lying areas without gravity-driven drainage systems are more vulnerable to surface water flooding. They also express their concern regarding the lack of a warning system in the UK [Coulthard and Frostick, 2010].

2.2 Hydraulic models

Previous work done to identify areas vulnerable to heavy rainfall mainly explores the possibilities of hydraulic models. Hydraulic models are used by engineers to simulate the flow of water in a specific area [Patra et al., 2016] [Chen and Djordjević, 2012]. Numerous hydrodynamic models have been developed, the performance of these models will differ depending on the area they are deployed. Surface flow velocities or the influence of drainage systems may or may not be taken into account depending on the model [Tesema and Abebe, 2020]. This thesis uses machine learning instead of hydraulic models to predict damage from heavy rainfall.

2.3 Machine learning methods

The task we are trying to have the computer perform is a classification task. Many classification algorithms have been developed to complete this task. Some examples are the Decision tree, k-nearest neighbors, or random forest. The random forest model was first proposed by Breiman

[2001]. The method takes a collection of trees and aggregates their result in an ensemble. On each node, a random subset of predictors is chosen. This randomness makes the model robust to overfitting. The random forest also has only two parameters and is not very sensitive to these values [Liaw et al., 2002]. This means that even an unoptimised random forest should be able to produce competitive results. Every machine learning problem requires the choice of the best algorithm with its optimal hyperparameter settings for the given data set. There is no universal method that works best on all possible data sets [Feurer et al., 2015]. Feurer et al. [2015] developed auto-sklearn. The pipeline comprises 15 classification algorithms, 14 preprocessing methods, and 4 data pre-processing methods. Measuring the performance of every possible combination with hyperparameter optimisation would be too computationally intensive. Instead, they make use of meta-learning. They gathered a large number of data sets and saved meta-features of these data set. When they have determined the best combination of methods for that data set a different data set with similar meta-features is likely to perform well when using the same methods. Auto-sklearn makes use of Bayesian optimisation. The overall performance can be improved by saving the models made during training and creating an ensemble from them. This results in a system that can create very efficient classification models in a relatively short amount of time. Automated machine learning has to visit a large number of models to determine the best configuration, so it will take much more computing time than training one single model. Feurer et al. [2015] compared auto-sklearn to other automated machine learning systems and found an improvement over auto-weka. Gijsbers et al. [2019] compared various AutoML tools using a random forest as a baseline. A time budget was set for 4 hours, which is much lower than the runtime Feurer et al. [2015] used. On binary classification tasks auto-sklearn achieved an equal or higher performance than the random forest on 19 out of 23 datasets [Gijsbers et al., 2019].

2.4 Statistical methods

Machine learning methods are often only marginally improved. To distinguish between random and non-random performance differences we use statistical tests. The use of a statistical test to determine if an improvement is significant is criticized by some researchers [Berrar and Dubitzky, 2019, Demšar, 2008, Drummond, 2006]. One of the objections that is made most often is that of the "curse of multiplicity" [Demšar, 2008]. The problem is that when we perform enough experiments a statistical significance can always be found. So it is important to only use a statistical test when this is appropriate. Demšar [2006] recommends the use of the Wilcoxon signed rank test for the comparison of two classifiers and the Friedman test for comparison of more classifiers over multiple data sets. These tests do not require a normal distribution which makes them more suitable for comparing a limited number of results. After the Friedman test has rejected the null hypothesis, the Nemenyi test can be used to find the classifiers that actually differ [Demšar, 2006].

2.5 Machine learning applied to pluvial flooding

Until now little research combines the investigation of rainwater damage with machine learning techniques. Lamers et al. [2020] has created a model using terrain height maps and precipitation measurements to predict Twitter messages referencing rainwater damage. The data on terrain

height was taken from the "Algemene Hoogtekaart Nederland" (AHN2) [Rijkswaterstaat", 2012] and the precipitation data comes from the Royal Dutch Meteorological Institute [Overeem, 2021]. The precipitation is measured on a 1000 x 1000 m area. If we use every terrain height value in this area we would have to include 40000 attributes. This approach involves a high vulnerability to noise. To reduce this Lamers et al. [2020] use an auto-encoder. Events of rainwater damage are approximated by using Twitter messages as a target attribute. A random forest is built upon these messages. The model achieves an accuracy of 57.9%, 62.6% precision, and 26.4% recall. We will use this code as a base and try to improve the model's performance by changing its target attribute, features and classifier.

3 Data gathering & methodology

Most of the work to create this model has gone into the creation of a data set from the different data sources. We will first explain the two different target attributes we used and how we obtained these. Then we answer the same questions for the height and rain features. Instances of rainfall occur infrequently compared to instances without. We have explored different sampling methods to balance the data set. The methods we used will be explained in this section. We then give an overview of the pre-processing steps which together create our data set. Finally, we describe the methods we have used to create the classification model.

3.1 Target attribute

The Twitter messages we used are the same Lamers et al. [2020] used to create a model. This data comes in the form of a JSON file which includes among other information for each Twitter message the date and time it was sent, the content of the message, and the coordinates it was sent from.

These messages are not a very reliable way to approximate real events of rain damage. For example, someone may decide to tweet about a flooded street they encountered days ago. The location, date, and time of the tweet will then differ from the actual event. A better target attribute could be p2000 alerts in the Netherlands. p2000 is the communication network of the Dutch emergency services. These alerts also come in a JSON file. This file contains, among other information, for each alert the data and time of the alert, the service required, a message describing the event and the province, place, and coordinates where the event occurred. There are some downsides to using this data as a target attribute as well. To start, all events in the data set required the assistance of emergency services. This means that minor incidents are not included in the data set. In addition, the data set contains instances of water damage. We cannot be certain that rainfall was the cause of this. We have taken some measure to limit the effect of this problem. The data set is filtered to only contain examples where at least 10 mm of rain has fallen in that area on the day of the alert. This threshold is a parameter that can be changed. We chose to use 10 mm, because this is the amount for which the Royal Dutch Meteorological Institute considers a day to be a "wet day". The threshold has been written into the code as a parameter so could have been changed to create different datasets for the experiments.

Every Twitter message or p2000 alert serves as a positive example. The location, date, and time

it was sent are used to determine the values of this example's attributes. For the implementation of the feature attributes it does not matter whether the coordinates or dates come from Twitter messages or p2000 alerts so for the sake of simplicity we will continue to refer to both as alerts.

3.2 Height features

The data on terrain height is taken from the "Actueel Hoogtebestand Nederland", more specifically AHN2 with a 5 meter resolution. [Rijkswaterstaat", 2012] The data comes in the form of TIFF images. These images cover an area of 5000 x 6250 m with a pixel giving a height value for a 5x5m square.

These images cannot be opened by a regular image viewer, but they can be viewed and edited by the GDAL library in Python. Using this library the image can be stored in a 2D-array where every position stores the values for a pixel in the image. From this information, we can determine the terrain height and geographical coordinates of this pixel.

The data on terrain height has to be turned into features that can be used by a machine learning model. If we use every pixel value as an attribute this would result in 1250000 attributes. This would be computationally intensive and make it difficult for the model to generalise. To reduce the number of attributes Lamers et al. [2020] used an auto-encoder. We have implemented a different approach. Instead of incorporating the entire image as attributes, we have decided to only use the data points closest to the location of the alert. Since the pixels of the height image cover a 5x5m square, it is not likely that the coordinates of an alert will exactly equal the coordinates of the pixel. Instead, we needed to find the pixel closest to the location of our alert. This could be done using a brute force approach by checking every pixel for its proximity to the alert, but for a 1000x1250 image and close to 6000 examples this would be very computationally heavy. To reduce the computation time we have implemented an algorithm inspired by the binary search algorithm. Instead of finding a value in a list, we now search for a pixel in 2-dimensional space. First, the space is divided into four rectangles and the location of their centers is determined. We then recursively continue to search the rectangle which center is closest to the location of the alert as can be seen in Figure 1. This reduces the complexity from $O(m * n)$ to a complexity of $O(^2log(m),^2 log(n))$ for an image of width m and height n. When we find the pixel closest to the location of the alert we take a 20x20 pixel area around it as height attributes as shown in Figure 2. This results in 400 height values for every example in the data set. The choice of a 20x20 pixel, or 100x100 meter area is an arbitrary one that could be optimised.

3.3 Rainfall features

For our data on rainfall we have used the rad_nl25_rac_mfbs_01h data set from the Royal Dutch Metreological Institude (KNMI). This data set contains raw climatological radar data measured every hour at a 1 km grid. [Overeem, 2021]

The KNMI data set spans a different area than the AHN_2 data set. This has been visualised for one message from the Twitter data set in Figure 3. The KNMI data set continues to be updated with new precipitation measurements. The data is stored in H5 files which can also be read using

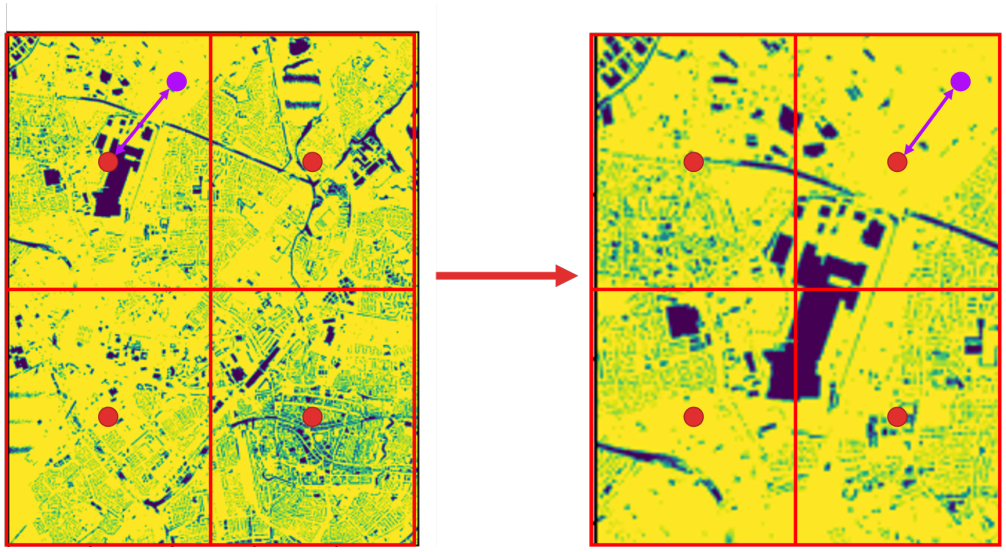

Figure 1: Graphical representation of the algorithm used to find the pixel closest to the alert for an example image from the AHN2_5m data set. The purple dot represents the location of an alert. The image is divided into four(red), the distance between the middle of these areas(red dots) and the alert (purple dot) is calculated. On the left side of the figure, the upper-left area is closest to the alert so the algorithm zooms in on this area on the right side of the figure. When applying the algorithm these steps would be reiterated until the area to zoom into is one pixel in size. This is when the pixel closest to the alert has been found.

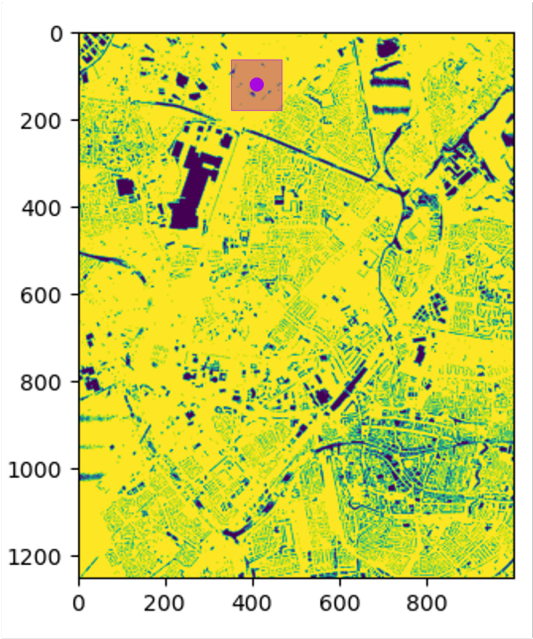

Figure 2: A 20x20 pixel grid is taken around the location of the alert to serve as height attributes. Example for an image from the AHN2_5m data set in Leiden.

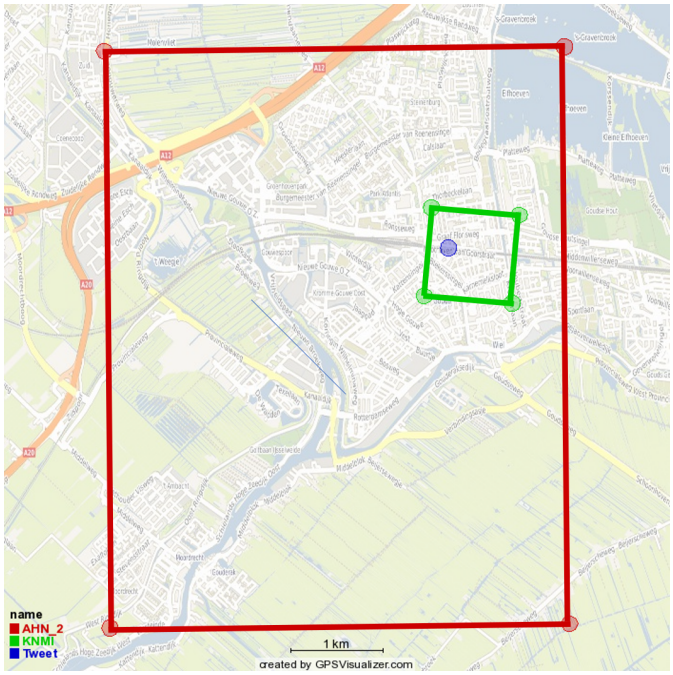

Figure 3: Map showing the area spanned by the target attribute, KNMI and AHN_2 data sets.

the GDAL library. Our first rain feature takes the amount of precipitation on the day of the alert within the 1km grid the alert falls into.

3.4 Sampling methods

Every alert, providing the amount of rain on that day exceeds the threshold, is taken as a positive example of rainwater damage. We made the assumption that in the absence of an alert there was no rainwater damage. In this case, every location in the Netherlands with precipitation exceeding the threshold on a given day could be considered a negative example. We then have many more negative examples than positive ones. This could have as a consequence that a model which always labels an example as negative can achieve a very high accuracy. To prevent this we aimed to create a data set with an equal number of positive and negative examples. This required sampling from the available negative examples. For all negative examples, we added the constraint that the day of the rain measurement must fall within the time frame of the alerts and that the amount of rain on that day exceeds the threshold of 10 mm.

The most straightforward way to do this is to sample for negative examples would be to take a random measurement of precipitation from the data set. A random location within the area where this measurement was taken will serve as the location to determine the values of the height features. We refer to this method as random sampling.

A potential pitfall of the random sampling method has to do with the land use in the Netherlands. Most of the land is used for agricultural purposes as illustrated in Figure 4. This means that when we sample a random location in the country this will likely be farmland. This is opposed to the alerts of rainwater damage that come from urban areas. This is a sampling bias that could have

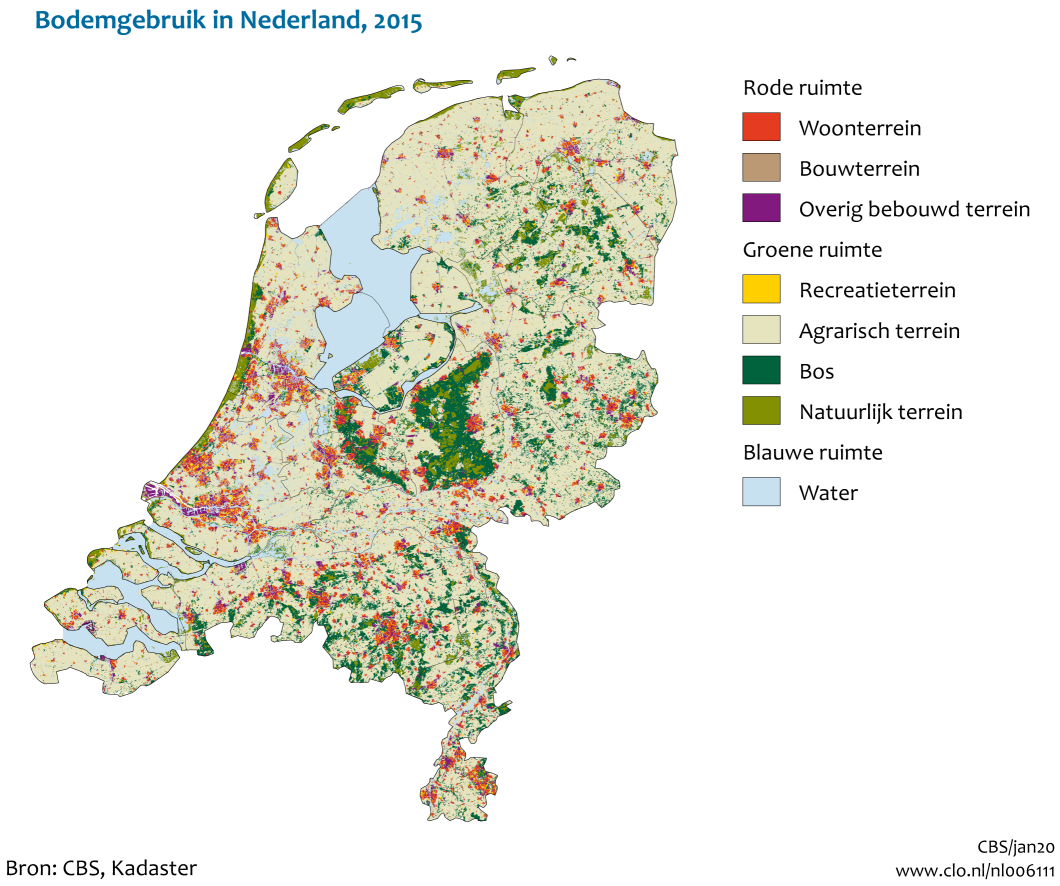

Figure 4: Map showing the land use per category in the Netherlands in 2015.

a large effect on the performance of the model. To prevent this sampling bias we made use of the Basisregistratie Adressen en Gebouwen (BAG). This register allowed us to sample from all addresses in the Netherlands. The data comes in the form of a JSON file containing the address and its Rijksdriehoek Coordinates. This a specific national coordinate system used by the land registry. These coordinates can be converted to longitude and latitude values so it is possible to locate them on the height images. A randomly chosen address serves as the location of a negative example. We then took a random example and use these measurements to create our rain features. We refer to this method as address sampling.

To make the classification task more realistic we have implemented a third sampling method. Up till now, the model has had the opportunity to distinguish areas with high or low risk of rainwater damage. The overarching question is then "Why do some areas experience damage from rainfall while others do not?". An interesting question related to this would be "Why do areas sometimes experience damage from rainfall and sometimes not?". Intuitively both lower and higher situated areas could experience damage from rainfall, but highly situated areas may need heavier rainfall to experience issues. To gain more insight into this problem we have implemented a dependant sampling method. We took the same location as a positive example and then found a random day without an alert. This positive and negative example form a pair that is put in either the training

or the test set. This is a dependant sampling method, it forces the model to distinguish between conditions that will or will not result in rain damage for the same location. The terrain height on its own does not predict the occurrence of rain damage, but can only be used in combination with the rain features. We will refer to this sampling method as equal-location sampling.

3.5 Engineering the dataset

The pre-processing steps can be run with six different settings. One for each combination of target attribute and sampling method. Each setup results in a slightly different data set. The complete workflow used to create these data sets is explained further. The KNMI and AHN2 data sets are very large so we had to take specific measures to limit the memory usage and computation time used when engineering these data sets. We first created a csv-file mapping the height images to geographical coordinates. The KNMI data is measured per radar grid. Each grid is given an x and y value to indicate its position in respect to the other grids. We also created a csv-file to map these radar coordinates to geographical coordinates. Due to the curvature of the earth, these radar grids and height images are not square in a 2D space. So, we had to save the coordinates of each corner to later determine if an alert falls within the grid or image. We saved the rain measurements in a csv-file and filtered them. Both the filtered and unfiltered measurements were stored for later use. The filtered measurements were later used to sample from. When we take the sum of rain from the hours before the alert we likely need to tap into the measurements recorded on the day before the alert. This means that even though the rain on the day before an alert may not exceed the threshold its data is still relevant. We saved the alerts in a csv-file and combine this with the previously saved information mapping radar grids to geographical coordinates to find the radar grids the alerts occurred in. We could then combine this with the filtered rain measurements to give each alert a rain attribute. Since there is only data for days with enough rain saved in the filtered data set this step automatically filters the alerts to only contain alerts where the rain that day exceeds the threshold. The next step differed depending on the chosen sampling method. When using random sampling or address sampling we first performed the sampling step and then determined the height features for the entire sample. When we used equal-location sampling the height features for the positive and negative samples are the same so we could save computing time by determining the height features for all positive examples and then copying these to the negative examples.

When we performed the pre-processing steps we dropped some examples in the data sets. We did this because some alerts did not have a proper location, the rain measurements on that day were incomplete or the amount of rain on that day did not exceed the threshold. The entire process is represented visually in Figure 5.

3.6 Classification model

We have trained two different models on the data set. The first is a random forest, without any hyperparameter optimisation and the second is an auto-sklearn model optimizing for accuracy. We first used 10-fold cross-validation to create a separate train and test set before we trained the model on the training set. We created a classifier using auto-sklearn with 10-fold cross-validation.

The auto-sklearn task had its time restricted to 1 hour with a time limit for individual runs of 5 minutes. We measured the performance on the test set with accuracy, precision, and recall. We took the average performance on all ten test sets as the performance of the model.

4 Experiments

We have conducted several experiments to answer the sub-questions. First, we compare the Twitter and p2000 data set to find which is most reliable to use as a target attribute. Then we compare data sets generated by the different sampling methods and the performance the model achieves on these data sets to find which sampling method best supports our classification task. We then compare the performance of a model using rain on the day of an alert as a feature to a model using terrain height features as well.

We first explored the p2000 and Twitter data sets to determine which would be most reliable to use as a target feature. We have compared the geographical distribution of the examples and the distribution of rain on the day of the examples. We have also compared the number of data points we lose when generating attributes and reducing noise in the data set.

The pre-processing methods result in six data sets, one for each combination of target attribute and sampling method. We have trained the auto-sklearn model on all six data sets. The feature subsets we initially used were the amount of rain on the day of the alert, the height features, and both of these combined. We chose to use the auto-sklearn model instead of the unoptimised random forest, because we expected from the literature that the auto-sklearn model would generally outperform the random forest.

We then compared the performance of the model using the different sampling methods. We have compared the performance on the different data sets with the p2000 alerts as a target, as attributes we used the amount of rain on the day of the alert and the height features.

Next, we tried to find out if the use of terrain height attributes improved the performance of the model. We have trained both the random forest and auto-sklearn model on all six data sets. To answer this question we only compared the performance on the data sets using the random or

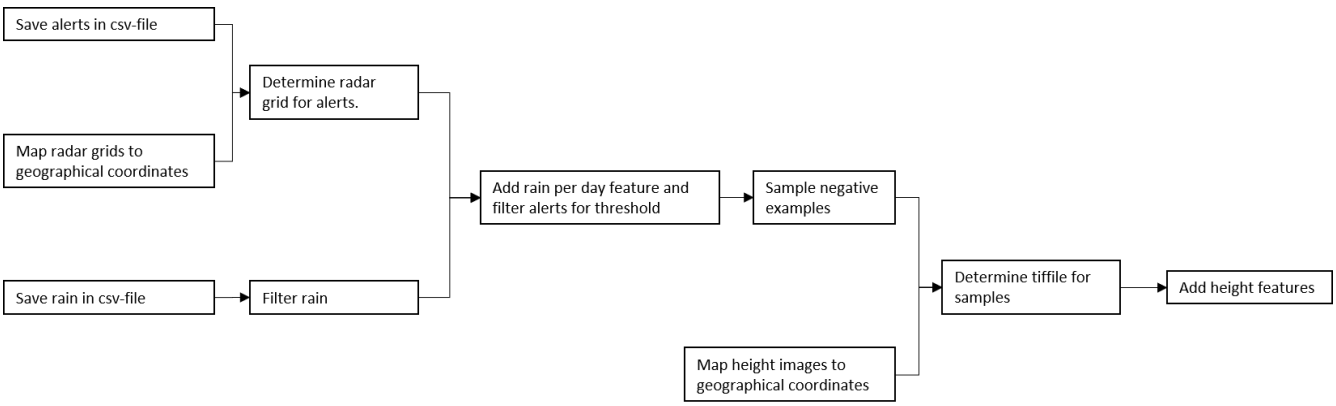

Figure 5: Visual representation of pre-processing steps used to create the data sets.

	Twitter	p2000
Timespan	okt 2010 - jan 2019	jan 2016 - feb 2021
Number of datapoints	5364	5800
Columns	date, latlon, text	date, latitude, longitude, message, prio, service, region, place, address, zipcode, province

Table 1: Table describing the p2000 and Twitter data sets.

equal-location sampling method with the p2000 alerts as a target. When we used the equal-location sampling method every positive example forms a pair with a negative one where both examples have the same height features. We have compared the performance of the auto-sklearn model using only the amount of rain on the day of an alert and using the height features in addition.

5 Results

5.1 Target attribute

We first explored both target data sets to determine their differences. Table 1 gives some information about the contents of both data sets. The Twitter data covers the time between October 2010 and January 2019 while the p2000 data covers January 2016 until February 2021. An important remark here is that almost all alerts from the p2000 data fall between 2016 and 2018. We have plotted the examples in the data sets on a map in Figure 6. Here we can see that the Twitter messages are more prevalent in the northeast of the country while the p2000 alerts are in the southwest. The southwest of the Netherlands is more densely populated than the northeast, making the p2000 data set more representative of urban areas. The sizes of the two data sets, without being filtered for a rain threshold, are comparable with 5364 Twitter messages and 5800 p2000 alerts. We also wanted to know if the Twitter and p2000 data sets report on the same event. When we combined the two data sets we found that there were only 862 cases where there was both a Twitter message and p2000 alert in a rain measurement area on the same day. Some of the Twitter messages were sent out by a fire station of the p2000 network, not taking these into consideration we are left with only 184 messages. This shows that there is little overlap between the two data sets.

The target data sets have been combined with the KNMI data to find the amount of rainfall on the day of an alert. During this procedure, we have lost some examples due to incomplete input or missing rain measurements. The original p2000 data set contains 5885 examples and the Twitter

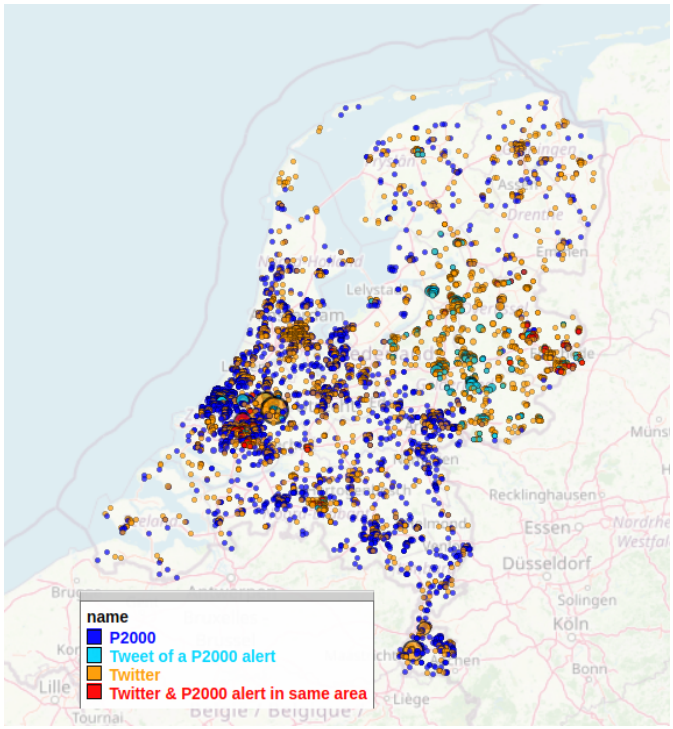

Figure 6: Map showing the location of all Twitter messages and p2000 alerts in our data set between 2016 and 2019.

data 6294. The data sets resulting from our pre-processing methods using the p2000 alerts as attributes contained 4259 examples while data sets using the Twitter messages contained 5224. So, with the p2000 alert, we lost 30% of our data while we only lost 17% on the Twitter data. Most of this loss can be attributed to missing rain measurements. On the p2000 data set, we lost 42 examples, because of missing information, on the Twitter data set this was 92 examples.

A shortcoming of both data sets is that alerts are for water damage and not rainwater damage in particular. To mitigate this shortcoming we filtered the data sets to only include examples with more than 10mm of rain on the day of the alert. 1455 examples in the Twitter data did not exceed the threshold, meaning we have discarded 29% of the data set in an effort to reduce noise. In the p2000 data set 455 examples, or 10%, did not exceed the threshold. The remaining data is plotted in Figure 7. We can see that the Twitter data contains more examples without rainfall and examples with extremely high rainfall. We expected to see more Twitter messages than p2000 alerts on days with less rain, because p2000 alerts are on a much higher escalation level. Messages and alerts on days with little rainfall may also be an indication of noise in the data sets. We have analysed the Twitter data manually to see if we could recognise noise in the data sets. Some examples of Twitter messages on a day without rainfall are:

“ #vansonvloeren is de expert voor het herstellen van #waterschade aan uw vloer. Bel ons voor advies of een #schaderapport op 0181-336944”, “http://t.co/JaKkP9dJ”

“Waterschade vaak grootste schade na brand. Amerikaans ondersoek”

“Alle zooi die na de overstroming in de kelder in huis stond weer naar de kelder verbannen op de

kerstspullen na. Die zijn binnenkort nodig!"

The first message is an advertisement, the second refers to the results of an American research study and the last one references an event of rainwater damage in June, while the tweet was sent in November. These are examples of noise that occur frequently in the data set.

Figure 7 shows the cumulative distribution function of rain for the Twitter and p2000 data sets. The frequency plot shows some very large peaks of messages in the Twitter data set. These messages do appear to reference events of rainwater damage. The messages were all sent on the same day in 2013 during one heavy rainfall event. As the p2000 data set does not cover 2013 this event is not represented there.

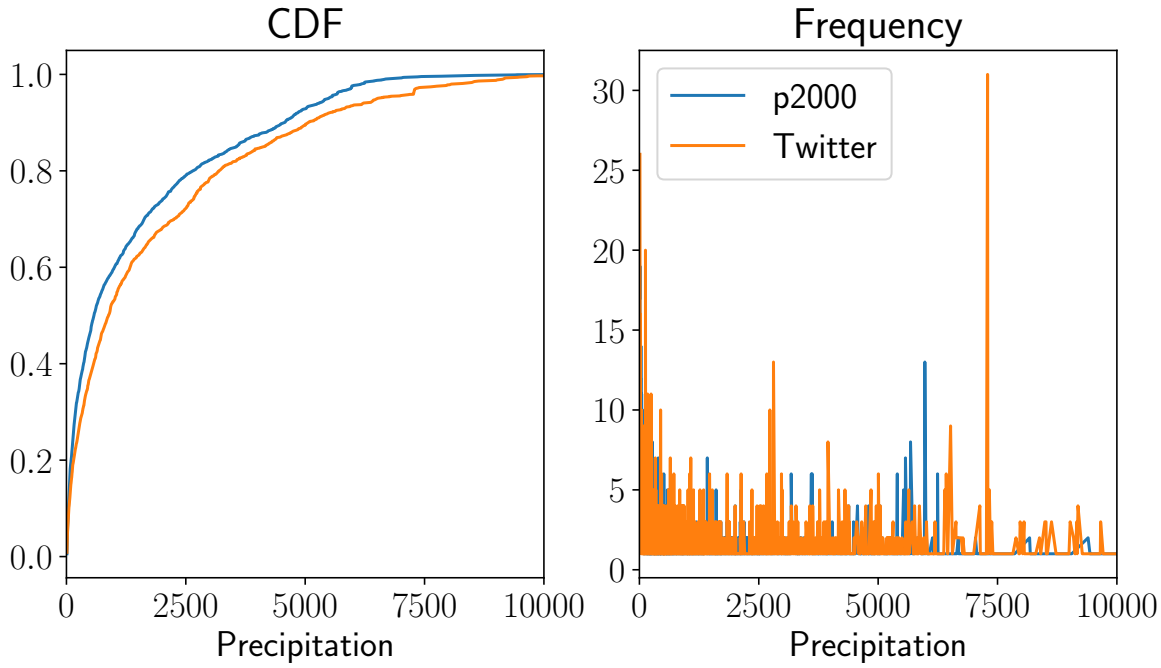

Figure 7: Distribution of rain for each target data set.

During the exploration of these data sets, we have seen that the Twitter data contains much more noise than the p2000 data set. We have performed experiments on all six datasets using autosklearn. Autosklearn is run on the training set of each of the 10 folds created with 10-fold cross-validation. The total time for each run is restricted to 1 hour. The time budget for each classifier autosklearn runs is 5 minutes. Table 2 shows the results of the auto-sklearn model using the rain on the day of the alert and height features. The results are given for each data combination of sampling method and target attribute. The performance is measured using the mean accuracy, precision, and recall for all ten test sets. The spread of these measures is visualised in Figure 8 and Figure 9. The model achieved a higher performance when trained on data sets using the p2000 data as a target. The spread of the performance measure was also smaller.

As we have judged the Twitter data set to contain more noise we have decided to use the p2000 data as a target in further experiments.

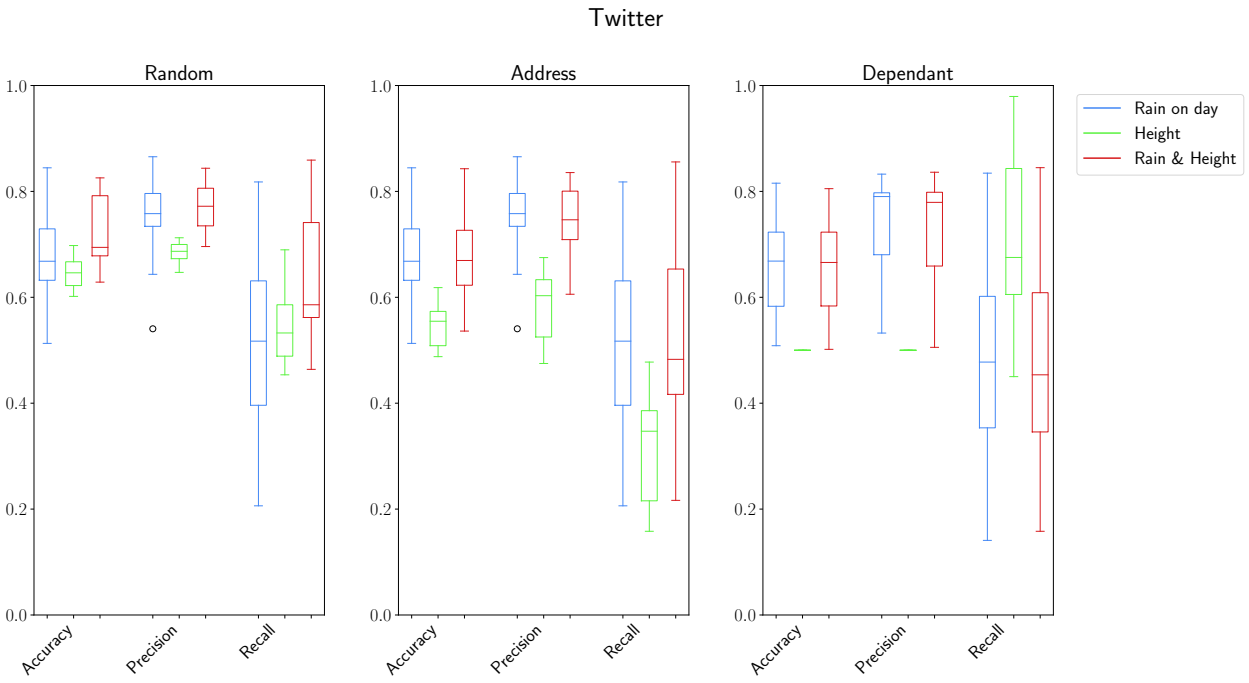

Figure 8: Results of experiments on data sets using Twitter messages as a target attribute.

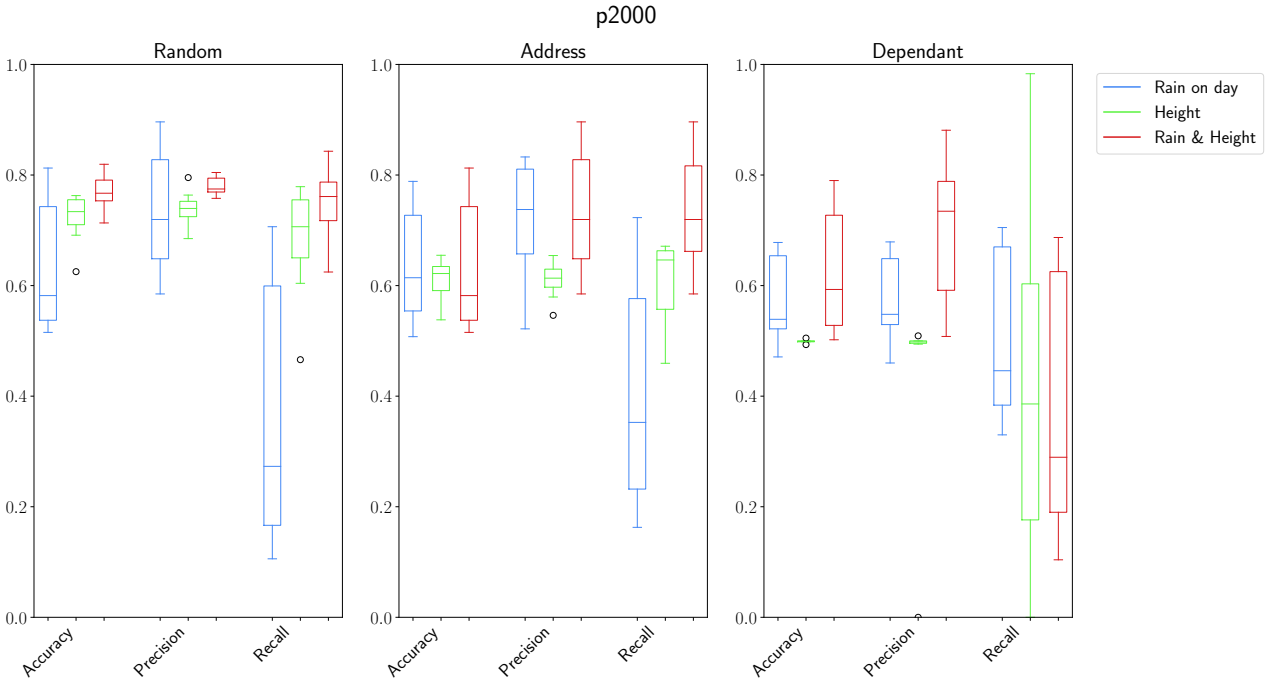

Figure 9: Results of experiments on data sets using p2000 alerts as a target attribute.

5.2 Sampling methods

		Random Sampling	Address Sampling	Equal-location Sampling
Twitter	**Accuracy**	**0.726 (±0.07)**	0.677 (±0.089)	0.656 (±0.098)
	Precision	0.772 (±0.048)	0.744 (±0.069)	**0.790 (±0.111)**
	Recall	**0.639 (±0.138)**	0.524 (±0.187)	0.478 (±0.215)
P2000	**Accuracy**	**0.770 (±0.032)**	0.690 (±0.064)	0.576 (±0.077)
	Precision	**0.780 (±0.017)**	0.716 (±0.048)	0.578 (±0.074)
	Recall	**0.752 (±0.063)**	0.620 (±0.135)	0.507 (±0.153)

Table 2: Table presenting the results of the auto-sklearn model using the rain on the day of the alert and height features for every combination of sampling method and target attribute.

Table 2 is also used to test whether the performance difference found between the sampling methods is statistically significant. We only compared the accuracy scores on the data sets using the p2000 data as a target. We compared the different data sets using the Friedman test.

H_0 : The mean accuracy for all three data sets is equal.
H_a : At least one mean accuracy is different.

If the p-value is below 0.05 the null hypothesis is rejected. The test resulted in a p-value of 0.00109. The null hypothesis has been rejected and we have accepted the alternative hypothesis that at least one mean accuracy is different. We then performed the Nemenyi test to determine which mean accuracies are different. The results of the Nemenyi test are presented in Table 3. At $\alpha < 0.05$ the only two statistically different means are between the random sampling method and the other two sampling methods.

	Random sampling	**Address sampling**	**Equal-location sampling**
Random sampling	1.000000	0.007169	0.002297
Address sampling	0.007169	1.000000	0.900000
Equal-Location sampling	0.002297	0.900000	1.000000

Table 3: Table presenting the results of the Nemenyi test on the mean accuracy of auto-sklearn model using the rain on the day of the alert and height features for every sampling method.

5.3 Height features

Our next experiment aimed to answer the question: Does adding terrain height attributes improve the model? We compared the performance of the auto-sklearn model using only the rain on the day of an alert as an attribute and both the rain and height attributes. We performed this experiment on the three data sets using the p2000 data as a target attribute. The results from this experiment are presented in Table 4 and visualised in Figure 9. As we compared different feature subsets, not different data sets, we used the Wilcoxon signed-rank test. The null hypothesis asserts that the medians of the two samples are identical. The resulting p-values can be found in Table 5. We again used $\alpha = 0.05$, all p-values fall below 0.05. This means the performance improvement from adding height attributes is statistically significant on all three data sets.

		Random	Address	Equal-location
Rain on day	**Accuracy**	**0.635 (±0,116)**	0.638 (±0.1)	0.576 (±0.077)
	Precision	**0.736 (±0.110)**	0.719 (±0.110)	0.578 (±0.74)
	Recall	0.372 (±0.245)	0.410 (±0.204)	**0.507 (±0.153)**
Rain on day & Height features	**Accuracy**	**0.770 (±0.032)**	0.690 (±0.064)	0.576 (±0.077)
	Precision	**0.780 (±0.017)**	0.716 (±0.048)	0.578 (±0.074)
	Recall	**0.752 (±0.063)**	0.620 (±0.135)	0.507 (±0.153)

Table 4: Table presenting the results of the auto-sklearn model using the rain on the day of the alert and the same attribute in combination with height features on the data sets for every combination of sampling method using the p2000 alerts as a target attribute

	Random sampling	Address sampling	Equal-location sampling
p-value	0.00694	0.01242	0.00932

Table 5: Table presenting the results from the Wilcoxon signed-rank test comparing the different feature subsets for the data sets with different sampling methods

6 Discussion & Limitations

6.1 Target attribute

During our exploration of the p2000 data set, we found that we had to discard a large number of examples during our pre-processing steps. We lost comparatively few Twitter messages when building our features. However, most of the unusable examples were the result of missing rain measurements. So, the amount of data lost during pre-processing is not an accurate measure of data quality.

The amount of noise is a large problem for both data sets. The Twitter data was originally generated by scraping all Twitter messages for certain keywords. During this procedure, there are several opportunities for noise to enter the data set. The Twitter message could use the keyword in a different context, as can be seen by our advertising and research study message. The time the message was sent does not have to align with the time of the event. A user can send a message about a previous event of rainwater damage or express concern about such an event happening in the future. The Twitter message may also not align with a rain damage event geographically. This can happen when a user sends a message recalling an event from a different location. Or when the user comments on a flooding event of another user.

When we use the p2000 data set we can be sure of a water damage event. Unfortunately, we can not be sure that rainfall was the cause of this event. There are a number of other causes for severe water damage, a water pipe can break, an aquarium can flood, or melting of snow can cause issues, to name a few examples.

We used a minimum threshold of 10 mm rainfall on the day of an alert to filter the data sets for events rain could be the cause of damage. We used 10 mm, because this is the number the Royal Dutch Meteorological Institute uses to define a wet day [KNMI", 2021]. It is highly unlikely to experience damage from rainfall on a day with less than 10 mm of rain. When we used this

threshold to filter our data sets we discarded 10% of our p2000 alerts and 29% of our Twitter messages. The first reason for this we have already described. The Twitter data contains more noise, so we expect more false positive examples. We do have to take into account that the Twitter data represents a lower escalation level than the p2000 data. In a case of minor damage, the fire department will not be involved. These minor events will be present in the Twitter data set, but not in the p2000 data. This can also be observed in Figure 7. This plot also shows that the Twitter data contains many examples of days with extremely high rainfall. Particularly a peak around 70.00 mm stands out. These Twitter messages all stem from the same day in 2013 where an extreme rainstorm caused a lot of damage. Our p2000 data does not cover 2013 so the same spike cannot be seen in this data set.

The models trained on the p2000 data achieved a higher performance than the models using the Twitter messages as a target. This can be, because of the extra noise in the Twitter data set. When the target data experiences more noise it does not reflect the actual relationship with the predictive features as well, making it more difficult for the model to train on.

6.2 Features

The use of height attributes significantly improves the performance of the model. This suggests that terrain height is related to the occurrence of rainwater damage. Even when we add these attributes we can see that the model relied much more on the rain features than height features when making its predictions. This preference can have multiple causes. The first would be that the relationship between rainfall and rainwater damage is simply stronger than the relationship between terrain height characteristics and rainwater damage. This is most likely when most rainwater damage occurs on failing roofs. When the structure of the roof cannot properly drain the rainwater it causes damage, independent of the terrain the building is on. A different reason could be that the height attributes are more difficult for the model to generalise over. The model receives 400 height attributes and only one rain attribute. When we feed a model a large number of attributes it becomes more difficult for the model to interpret the information.

A limitation of the height features is that they are not optimised. The choice of a 20x20 grid around an alert is an arbitrary one. This means that the features we have used to train these models may not be the most predictive features we could generate from the height data.

The most likely explanation, however, can be found in the distribution of rain for positive and negative examples. The cumulative distribution function for rain is shown in Figure 10. The figure shows the distribution for positive and negative examples using the different sampling methods is shown in Figure 10. We can see clearly that the negative examples have less rainfall than the positive ones. This means the classifier can easily split on this attribute to achieve a high accuracy.

6.3 Sampling methods

During our experiments, we could see that the data sets generated with random sampling performed better than those generated with address-based sampling and equal-location sampling. Our reason for implementing a more complex sampling method was that the majority of land in the Netherlands

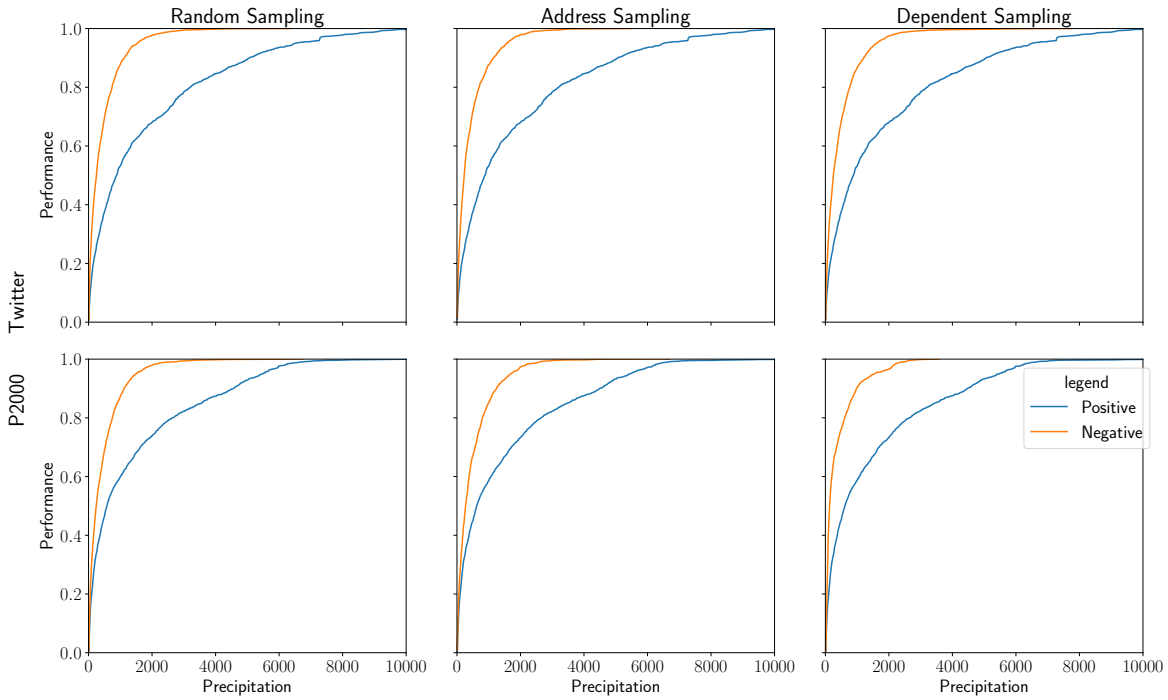

Figure 10: Distribution of rain for each target data set.

is used for agricultural purposes. This can be seen in Figure 4. The geographical distribution of the samples can be seen in Figure 11. The dots are positive examples of rainwater damage and red dots negative ones. When we use equal-location sampling the locations of positive and negative examples are equal so we only plot the red dots. We can see from Figure 11 that negative examples selected using random sampling are spread evenly across the country. When we use address-based sampling we can see red clusters of negative samples around the large dutch cities. The positive and negative examples seem quite similarly distributed. This explains results from the Nemenyi test.

Urban areas have different terrain height characteristics than agricultural areas. When most positive examples are from urban areas and most negative examples are not, it allows the model to select based on the difference between urban and non-urban areas, not purely their height. This explains why we can see in Figure 8 and 9 that the models using only height attributes trained on data sets generated with the randoms sampling achieve a higher performance than models trained on data sets generated with the address sampling. When we introduce equal-location sampling the height attributes no longer have any predictive power on their own. After all, the training- and test sets contain one positive and one negative example for all height features. The height features only have predictive power when combined with rain measurements. So, 50mm of rain in a day may cause issues in lower situated areas and not in higher ones. As we make the sampling method for negative samples more complex the classification task itself becomes more complex, so we expect to see a lower performance.

We found during our experiments that adding height attributes to the model resulted in a statistically significant improvement in its mean accuracy. The intuition behind this result would be that water flows down, so lower lying areas are more prone to flooding from heavy rainfall. This relationship

Figure 11: Geographical distribution of examples for each target data set.

causes the inclusion of height attributes to improve the model.

We achieved the highest performance on the data set using the p2000 data as a target and the random sampling method. This model achieved a mean accuracy of 77%, mean precision of 78%, and mean recall of 75%. We expected this performance as this is the data set with the least noise and the sampling method resulting in the easiest classification task.

7 Conclusions & Further research

7.1 Conclusion

In this bachelor thesis, we have used machine learning techniques on data from multiple sources to predict damage from rainfall in urban areas. We aimed to answer the following research questions: Which data set is most reliable to use as a target data set? Which sampling method best supports the classification task? Does adding terrain height attributes improve the performance of the model?

We have compared a Twitter data set containing messages referencing rainwater damage to a data set containing emergency alerts after events of water damage. The goal was to find which data set would be most reliable to use as a target attribute for our model. We found the emergency alerts to contain less noise than the Twitter messages. So, the p2000 alerts from the dutch emergency network would be most reliable to use as a target attribute.

We used three different methods to sample negative examples. The first method takes entirely random examples. The second method uses a list of Dutch addresses to sample from and the final

method uses the same location as a positive example, but on a different day. Random sampling achieves the highest performance. This sampling method creates a data set containing a large number of examples from non-urban areas. These areas have different height characteristics making it easier for the classifier to separate the positive from the negative examples. The negative examples generated by the address-based method are from urban areas and overlap more with the locations of rainwater damage alerts. Examples created using the equal-location sampling method match the location of positive examples perfectly. To answer our second sub-question, while the random sampling method achieves the highest performance, we find that address-based and equal-location sampling methods better support the classification task.

We have introduced height attributes to the model by taking height measurements from an area of 100x100m around an example. With height measurements taken every 5 meters, this resulted in 400 height attributes. When we compare the performance of the model using only rain as a feature and to the performance when we include height attributes as well, we find an improvement of the model's performance.

This bachelor thesis applies existing machine learning techniques to a new domain. We have done this by engineering features from terrain height images and precipitation measurements. We have also developed proper sampling methods for machine learning tasks applied to urban areas. We have provided the domain with more insights into the contribution of terrain height and rainfall to the occurrence of rainwater damage in urban areas. Both are can be used when predicting rainwater damage, using them in combination yields the most accurate results.

7.2 Further Research

In further research, the model could be improved in several ways. The height features used could be improved. This could be done by optimising the size of the area around an alert to use as features. Alternatively, a different technique could be used to pre-process the data, for example, a convolutional neural network. The target data set still has a significant shortcoming by not reflecting rainwater damage specifically. Further research could contain the search for a higher quality target data set or the development of a more advanced method to separate instances of water damage from instances of rainwater damage. Further research could also use data from other sources to add new features to the data set. Some suggestions for these features would be sewage system or building characteristics.

References

Daniel Berrar and Werner Dubitzky. Should significance testing be abandoned in machine learning? *International Journal of Data Science and Analytics*, 7(4):247–257, 2019.

Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

Albert S. Chen and Slobodan Djordjević. Pluvial flood modelling and hazard assessment for large scale urban areas. In *10th International Conference on Hydroinformatics*, 2012.

TJ Coulthard and LE Frostick. The hull floods of 2007: implications for the governance and management of urban drainage systems. *Journal of Flood Risk Management*, 3(3):223–231, 2010.

G Dekker, T Nootenboom, L Locher, and MH Spekkers. Van last naar les: Hoe publiek-private samenwerking de regenwateroverlast voor inwoners, woningeigenaren en klanten kan verlagen. een analyse van schadegegevens en de factoren die van invloed zijn op regenwaterschade. 2016.

Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, 7:1–30, 2006.

Janez Demšar. On the appropriateness of statistical tests in machine learning. In *Workshop on Evaluation Methods for Machine Learning in conjunction with ICML*, page 65, 2008.

Chris Drummond. Machine learning as an experimental science (revisited). In *AAAI workshop on evaluation methods for machine learning*, pages 1–5, 2006.

Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Springenberg, Manuel Blum, and Frank Hutter. Efficient and robust automated machine learning. In *Advances in Neural Information Processing Systems 28*, pages 2962–2970. Curran Associates, Inc., 2015.

Pieter Gijsbers, Erin LeDell, Janek Thomas, Sébastien Poirier, Bernd Bischl, and Joaquin Vanschoren. An open source automl benchmark. *arXiv preprint arXiv:1907.00909*, 2019.

KNMI". KNMI - Regenintensiteit, 2021. URL https://www.knmi.nl/kennis-en-datacentrum/uitleg/regenintensiteit.

C. Lamers, J. van Rijn, and T. Beenen. Data science-technieken om regenwateroverlast in stedelijk gebied te voorspellen. *H2O Waternetwerk*, 2020.

Andy Liaw, Matthew Wiener, et al. Classification and regression by randomforest. *R news*, 2(3): 18–22, 2002.

A. Overeem. Precipitation - 1 hour precipitation accumulations from climatological gauge-adjusted radar dataset for The Netherlands (1 km) in KNMI HDF5 format - KNMI Data Platform, 2021.

Jagadish Prasad Patra, Rakesh Kumar, and Pankaj Mani. Combined fluvial and pluvial flood inundation modelling for a project site. *Procedia Technology*, 24:93–100, 2016. ISSN 2212-0173. International Conference on Emerging Trends in Engineering, Science and Technology (ICETEST - 2015).

Rijkswaterstaat". Actueel Hoogtebestand Nederland 2 (AHN2), 2012.

Viktor Rözer, Meike Müller, Philip Bubeck, Sarah Kienzler, Annegret Thieken, Ina Pech, Kai Schröter, Oliver Buchholz, and Heidi Kreibich. Coping with pluvial floods by private households. *Water*, 8(7):304, 2016.

Matthieu Spekkers, Viktor Rözer, Annegret Thieken, Marie-Claire ten Veldhuis, and Heidi Kreibich. A comparative survey of the impacts of extreme rainfall in two international case studies. *Natural Hazards and Earth System Sciences*, 17(8):1337–1355, 2017.

Janez Sušnik, Clemens Strehl, Luuk A Postmes, Lydia S Vamvakeridou-Lyroudia, Hans-Joachim Mälzer, Dragan A Savić, and Zoran Kapelan. Assessing financial loss due to pluvial flooding and the efficacy of risk-reduction measures in the residential property sector. *Water Resources Management*, 29(1):161–179, 2015.

Dejene Tesema and Birhanu Abebe. A review of flood modeling methods for urban pluvial flood application. *Modeling Earth Systems and Environment*, 6, 09 2020.

A Appendix 1: Rain features on an hourly basis

We have explored the possibility of adding rain features per hour instead of per day. We then looked at the rain per hour and used the sum of the precipitation in the hours preceding an alert as features. Assumptions we made earlier in the project resulted in a framework where the performance of the model was for daily rain features was not comparable to a model using hourly rain features. The assumptions we made caused the results of the model using hourly-based features to be unreliable. Adjusting these assumptions would have required the entire pre-processing stage of the data to be redone. This was no longer feasible to do within the thesis. The methods we have used during this thesis, results from the experiments, an explanation of the results, and possible ways to adjust the assumptions to be able to achieve more interpretable results are described in this appendix.

We use the same data from the KNMI data set we previously used to determine the amount of rain on the day of the alert. Now, we use the sum of precipitation in the hours preceding an alert as features. We use measurements up to 22 hours preceding the alert, resulting in 22 features. For an alert sent out on 16:15, the first rain sum would be the amount of rain between 15:00 and 16:00, the second between 14:00 and 16:00, etc. These attributes can be used to find which time frame best predicts rainwater damage.

For each run, we used one attribute. So the total amount of rain in 1 hour preceding the alert for the first run and the total amount of rain in the 2 hours preceding the alert for the second run. We did this up till the sum of rain during the 22 hours preceding an alert. It is important to point out that the attribute taking the sum of rain during 22 hours before the alert is quite different from the attribute taking the total amount of rain on the day of the alert. When an alert is sent earlier in the day the former will take precipitation measurements from the day before the alert as well. The earlier in the day an alert is sent the less similar the value of the two attributes will be. We have trained the model on every attribute, so 22 times. In every run, we used 10-fold cross-validation to split the data set and then train the model on the training set using the random forest. Using auto-sklearn would have required too much runtime. We expected the random forest to still result in a good performance, while taking significantly less time to train.

The mean accuracy, mean precision, and mean recall of the model trained on each feature is presented in Figure 12. We can see that the mean recall between 5 and 15 hours before the alert is higher than the mean accuracy and mean precision. This is different from all other experiments we have run, where the mean recall fell below the mean accuracy and precision. Up to 15 hours before an alert the mean accuracy falls around 0.5, this is worse than we expected. When the mean accuracy of a model falls around 0.5 it is about as good at predicting the target attribute as a random guess would be. When we move to features reaching more than 20 hours back we achieve a similar performance value as the random forest model trained on the amount of rain on the day of the alert.

When conducting experiments performance using features covering a shorter time span our model performed much worse than expected. Upon further exploration of the data set, we found the explanation for this. All examples in the data set have been filtered to have at least 10 mm of rainfall on the day of the alert. We did this to provide some filter for cases where rain could be the cause of water damage. This filter does not take the time of the alert into account. So an example can enter the data set when the alert was sent out at 11:30, while the KNMI only recorded rainfall

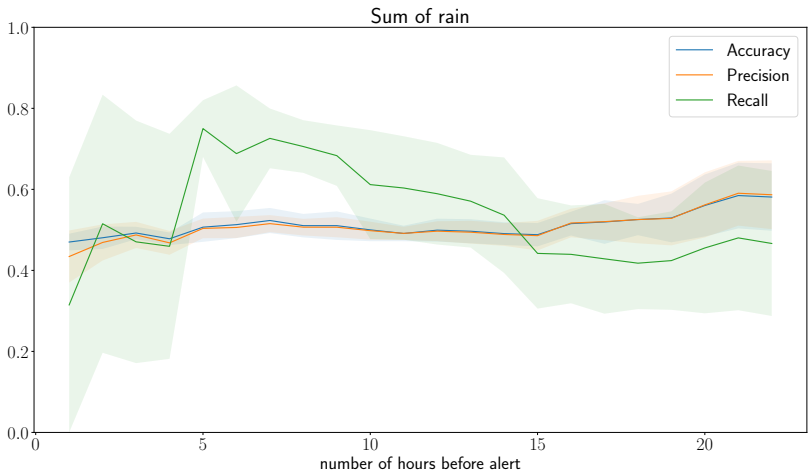

Figure 12: Performance of features reflecting the total amount of rainfall during any number of hours preceding an alert.

between 14:00 and 18:00. In this example, the rainfall on that day will exceed the threshold, but the rainfall in the hours leading up to an alert would be 0. This phenomenon is most prevalent in features spanning only a short time before an alert. The number of instances without any rain for each attribute are plotted in Figure 13. The proportion of examples without rainfall are similar for negative and positive examples, so the data set contains about as many negative as positive examples without rain. These examples are impossible to distinguish from each other. When the majority of examples in the data set have a zero-value for that rain attribute it becomes clear why the model performs about as well as a random guess. This experiment highlights again the downside of using a data set containing instances of water damage and not instances of rainwater damage specifically. The simple filter we used on the data proved insufficient to mitigate the shortcoming of the target data set.

When do not take into account the zero-values for each feature we would still expect features over a longer timespan to outperform the features calculated over a short amount of time. Figure 14 shows the distribution of rainfall for the features calculated over 5, 10, 15, and 20 hours before the alert. As we go further back the distribution of positive and negative examples starts to differ more, making it easier for the model to differentiate the classes.

To create data set which results in more comparable performance on different time span we would have to apply a different filter to the target data. We could filter the data set for examples where the sum of rain for all features exceeds a certain threshold. This filtering method may result in the loss of valuable samples. If heavy rainfall in 3 hours is enough to cause rainwater damage the example should not be excluded because there has not been rainfall 18 hours before the alert. If we only filter on the feature we are using the input data set would differ from feature to feature, making their performance incomparable. This why filtering on all features would be preferable. To be able to compare these results to the baseline achieved using the rain on the day of the alert as a feature this data set would also have to be filtered on the hourly rain sums.

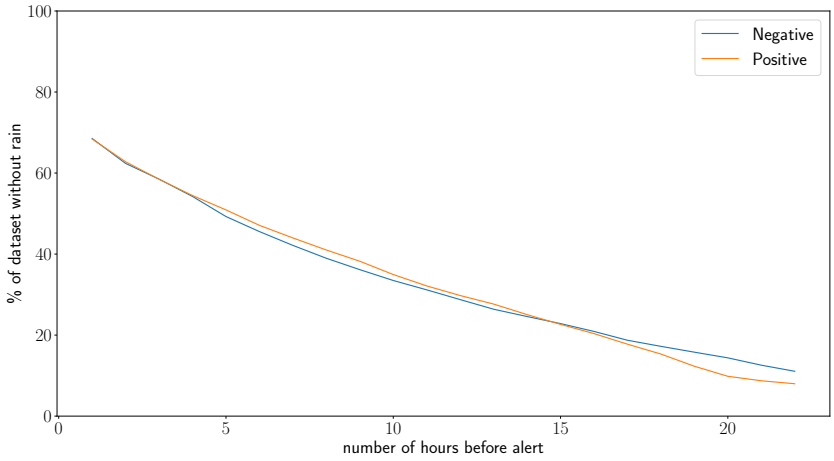

Figure 13: The number of instances without rainfall for both negative and positive examples, as a percentage of the number of negative/positive examples, for all hourly rain features.

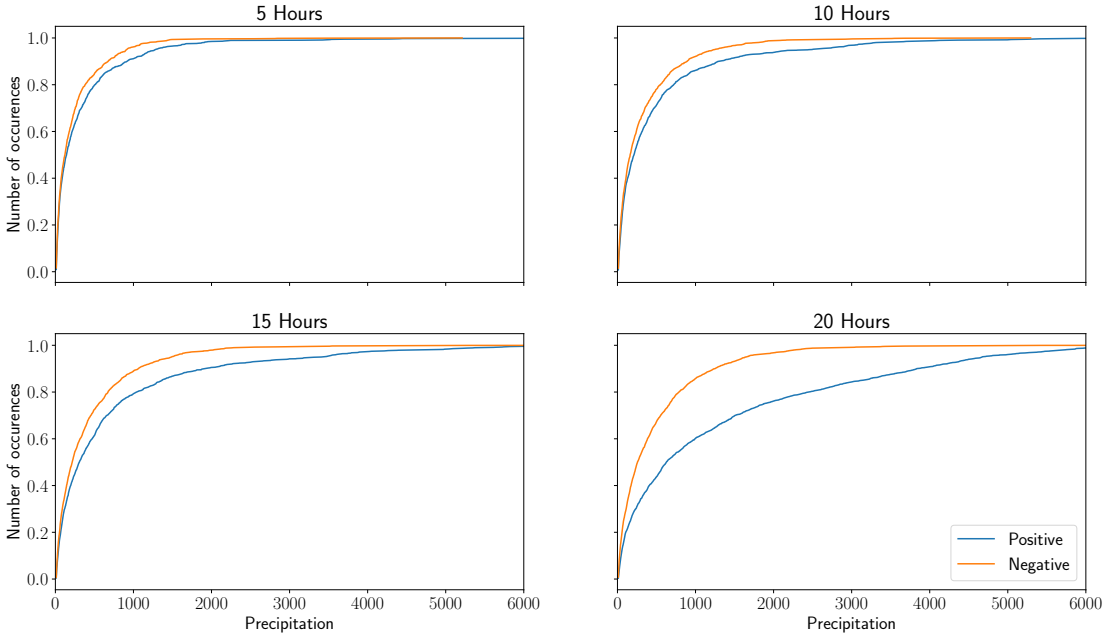

Figure 14: Distribution of rain for attributes measured on an hourly basis.