Using Object Detection to Classify Rain for TCE Group

Boaz Beukers, Denny Smit, Maxim Vitalis, Rhuben Vlug

Februari 3rd 2020

1 Abstract

Radar systems installed on three major windparks in the North Sea collect valuable data for ecologists studying bird movement. These radar systems are very sensitive meaning that not only birds are being detected but also rain. The classification of the radar system is unable to accurately distinguish between birds and rain, making it necessary for scientists to manually look at the data and filter out the raintracks. Time that could be spent on valuable ecological research is being unnecessarily wasted. This research focuses on how machine learning can classify the data. Through the implementation of a Scikit-Learn Random Forest Classifier, the radar data can be accurately (>99%) split, making the data from the radar systems much more useable and appealing for the purpose of birdresearch.

2 Table of contents

1 Abstract 1

2 Index 2

3 Problem Description 3

4 Data 4

4.1 Challenges 4

4.2 Preprocessing 5

5 Methode 6

5.1 K-means 6

5.2 Bayes Classifier 7

5.3 Random Forest: Overview 8

5.4 Random Forest: Details 9

6 Results 9

7 Conclusion 10

8 Discussion 11

8.1 Future work 11

9 References 13

3 Problem Description

Early january 2020, Jens van Erp, a PhD candidate studying wildlife in the North Sea in collaboration with TCE Group and their representative Ji Qi, asked us to help find a solution for excessive clutter in his large scale radar dataset. Three major windparks in the North Sea are fitted with specialised bird radar stations, which are used to gather track data from all kinds of different birds that enter the airspace. These tracks are vital to monitor bird movements and learn about the ecological impact of these offshore windparks on wildlife.

The data was gathered using a specialised bird radar, which was able to crudely perform initial classification, but unable to reliably split birds from rain. In the first image, figure 1, the information of the radar is shown. Each colour indicates a specific object, like a bird. In the second image below, figure 2, the problem is illustrated. The radar will classify this purple group of objects as a flock, while it is in fact rain. This is the cause for clutter in the dataset which makes it an impure dataset. The clutter, among others, consists of rain, airplanes and reflections of the sea. Because of this clutter, the dataset, as directly outputted by the Robin Radar, is unusable to researchers During the project, our goal was to filter raintracks from birdtracks

Van Erp showed his way of splitting the rain tracks from the non-rain tracks but mentioned himself that this was an inaccurate and time consuming way to do so. He asked us to create a more elegant algorithm, which would extract the rain data more accurately and faster.

Using methods common in artificial intelligence and data science, it is our task to find non-bird tracks. These tracks should be filtered from previously gathered data as well as new data, that is constantly being gathered in the three windparks.

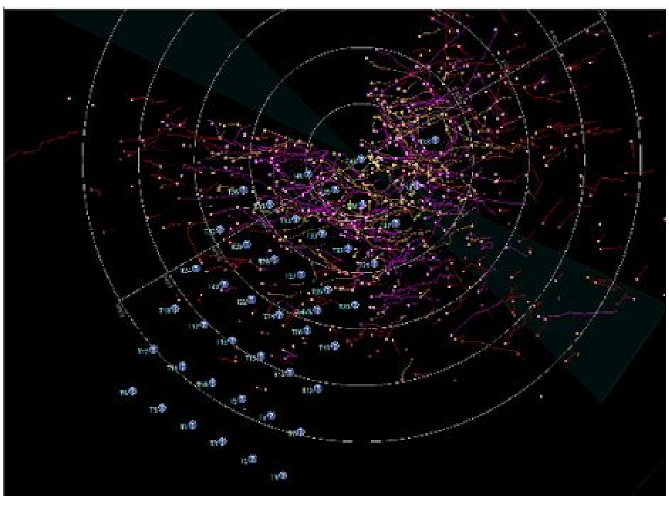

Figure 1: Radar Imagery

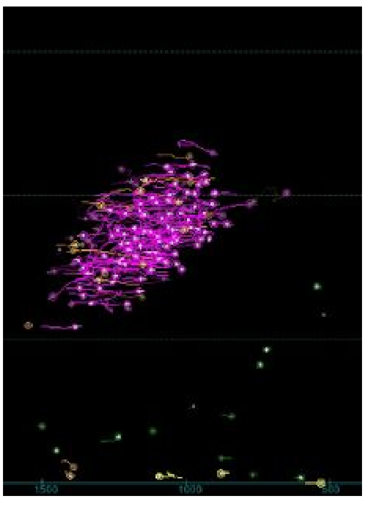

Figure 2: Rain Shower

4 Data

The available data consists of all tracks found by the Robin radar over the course of last year, as well as a manually filtered list, mentioned earlier, of tracks that are highly likely to be rain.

The tracks found by the radar originally contains 19 columns/features (figure 3), of which 14 were initially made available to us for use. The 5 features that were not included were described to us as being unsuitable to work with, unreliable or unusable without more in depth knowledge.

```
id,timestamp_start,timestamp_end,classification_id,rho_diff,phi_diff,theta_diff,tracktype,distance_travelled,airspeed,score,nr_of_plots
98262756,2019-03-01 00:00:00,2019-03-01 00:00:38,3,18.4264,-1.02758,0,RaAz,179.383,8.18166,0.916404,22
98262755,2019-03-01 00:00:01,2019-03-01 00:00:37,8,138.607,2.91804,0,RaAz,420.876,11.849,0.810779,20
98262771,2019-03-01 00:00:02,2019-03-01 00:00:50,8,431.62,0.79884,0,RaAz,843.907,15.7788,0.860547,25
98262753,2019-03-01 00:00:02,2019-03-01 00:00:36,8,103.807,0.894319,0,RaAz,406.612,13.3837,0.835145,18
98262735,2019-03-01 00:00:02,2019-03-01 00:00:16,8,308.518,0.231349,0,RaAz,313.626,27.3043,0.7253,9
98262751,2019-03-01 00:00:03,2019-03-01 00:00:35,7,220.717,0.503308,0,RaAz,590.266,18.1159,0.774333,16
98262744,2019-03-01 00:00:04,2019-03-01 00:00:27,8,535.237,2.04821,0,RaAz,578.969,18.5066,0.705314,11
98262758,2019-03-01 00:00:05,2019-03-01 00:00:39,8,616.666,-0.747124,0,RaAz,763.813,26.9484,0.861444,20
98262749,2019-03-01 00:00:05,2019-03-01 00:00:31,7,93.626,-2.27634,0,RaAz,332.712,14.2923,0.734945,12
98262764,2019-03-01 00:00:06,2019-03-01 00:00:44,8,640.94,3.07339,0,RaAz,719.957,15.2534,0.742111,15
98262750,2019-03-01 00:00:06,2019-03-01 00:00:31,8,405.193,3.02914,0,RaAz,451.949,13.4008,0.797406,15
98262743,2019-03-01 00:00:06,2019-03-01 00:00:27,8,60.3326,-0.413409,0,RaAz,212.96,13.7061,0.788089,12
98262759,2019-03-01 00:00:08,2019-03-01 00:00:39,8,236.273,1.9272,0,RaAz,470.196,14.5854,0.731023,13
98262746,2019-03-01 00:00:08,2019-03-01 00:00:30,8,613.558,0.0380588,0,RaAz,644.379,33.2214,0.722696,10
98262765,2019-03-01 00:00:09,2019-03-01 00:00:44,8,525.754,0.848752,0,RaAz,687.131,18.7158,0.790401,18
98262745,2019-03-01 00:00:09,2019-03-01 00:00:28,8,419.752,-0.721269,0,RaAz,461.077,28.3418,0.716855,11
98262761,2019-03-01 00:00:10,2019-03-01 00:00:41,8,605.885,-0.771116,0,RaAz,655.796,26.1981,0.783611,17
98262754,2019-03-01 00:00:10,2019-03-01 00:00:36,2,147.251,-1.327,0,RaAz,153.621,10.8968,0.847538,15
98262775,2019-03-01 00:00:14,2019-03-01 00:00:55,8,86.5932,-1.86481,0,RaAz,606.362,14.2685,0.821876,18
98262760,2019-03-01 00:00:15,2019-03-01 00:00:40,7,162.649,0.436459,0,RaAz,199.535,10.0905,0.787256,14
98262763,2019-03-01 00:00:16,2019-03-01 00:00:43,8,149.339,-0.111727,0,RaAz,318.137,13.8969,0.713203,13
```

Figure 3: Raw dataset

Furthermore, we decided to not use the trajectory feature, despite knowing this feature could be very promising. We decided on this due to the limited time, the level of difficulty to use this feature in a meaningful way and the fact that our results without the use of this feature already showed promise.

All tracks are labeled with an id and a timespan (begin and end), during which the radar was able to follow that object.

4.1 Challenges

The data available to use was delivered via online download of large .csv files. Because of the sheer size of the dataset, we were unable to work on all the data at the same time. After having a meeting with Van Erp to ask his opinion on the matter, we collectively agreed that we should focus on the month September as this month contains the most amount of rain.

Another challenge we faced concerning the data, was the substantial imbalance in the data. The dataset contained a lot more non-rain samples than rain samples (see figure 4). This meant that we had to revise our way of scoring the algorithm. Before, we scored our algorithm, as is common practise, by comparing the values predicted by the algorithm, to the actual labels of the data entry and seeing to what extent they matched. We called this score: similarity. Now, we score each algorithm by looking at the amount of false positives and false negatives. (Explained further at page 6: K-means)

However we prefered to minimize the false negatives, meaning we the amount of rain data being classified as non-rain data. We did this after Jens told us he prefered the program to over classify rain. We also hypothesized that the amount of false positives could never be zero, because Jens asked us to find new rain data(that the crude way of classifying missed).

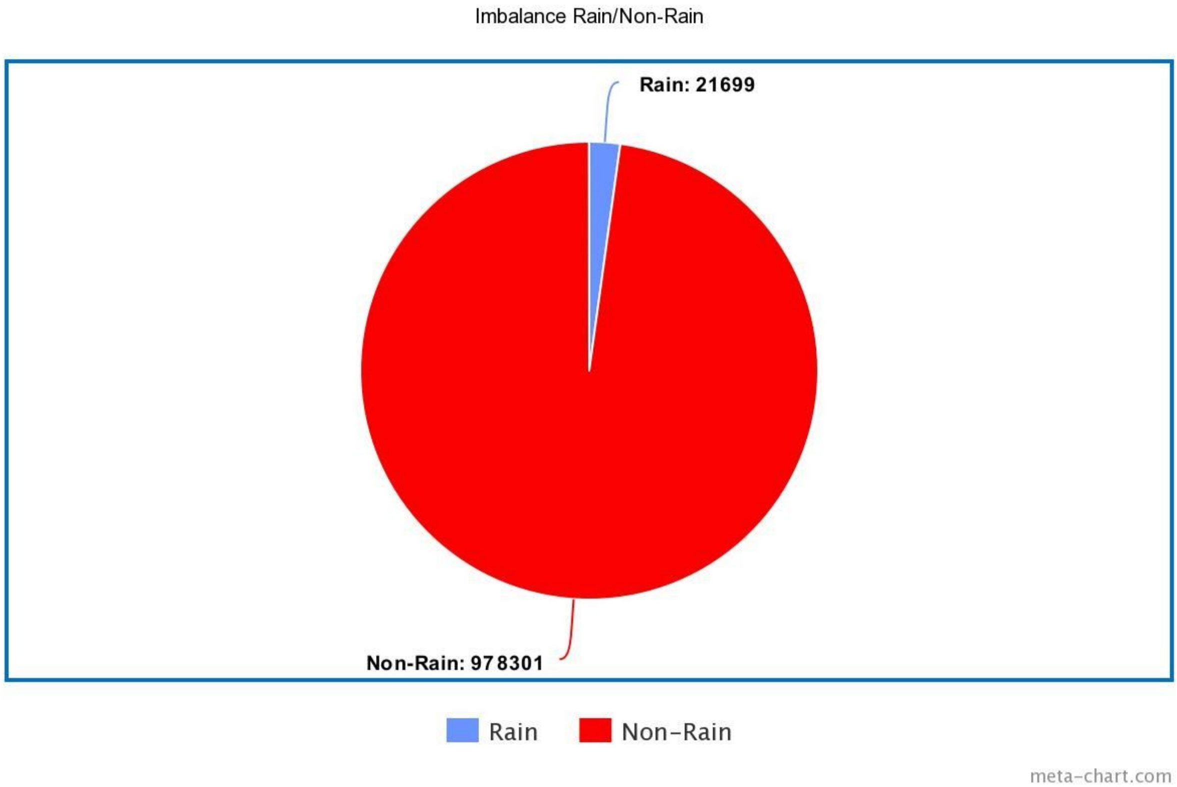

Figure 4: Imbalance in dataset

4.2 Preprocessing

The dataset, as delivered, was not ready for further analysis and classification. To start, the dataset containing the tracks was converted to a Pandas Dataframe, where each column represented one of the features.

Features that were dropped are the timestamps and the tracktype. The reasoning behind this is that, within the time available to us, we could not extract information from the timestamps in way that was relevant to us and powerful in classification. Furthermore, the tracktype feature does not contain information about the track itself but only tagged each track as being collected from one of the three identical radar stations.

In addition, we created an extra feature using that data provided in the list of rain id's. The id's in this list were filtered to only containing the month September and then matched against the large dataset to see which tracks, should be rain. To contain this new piece of information, a new column was created that labelled each track with a binary classification containing either ‘0’ for no rain, or ‘1’ indication rain (see figure 5).

id	rho_diff	phi_diff	theta_diff	distance_travelled	airspeed	score	nr_of_plots	rainbinary
115262749	3.72443	1.605680	0.000000	29.49290	17.73100	0.764462	9	0
115262750	18.43010	1.733310	0.000000	320.91600	11.72860	0.879623	14	0
115262752	17.39660	2.947380	0.000000	251.08400	16.06810	0.881707	14	0
115262753	125.90300	-0.584628	0.000000	153.63200	21.68810	0.845656	12	0
115262754	52.20800	-1.662730	0.000000	277.21900	22.03570	0.935440	18	0
115262757	12.51380	1.460070	0.000000	90.53860	14.60890	0.883251	13	0
115262758	85.94210	-1.224510	0.000000	298.37700	21.10850	0.726981	12	0
115262759	8.33019	-0.959956	1.067500	6.81861	20.63400	0.672848	8	0
115262760	160.33500	-0.444997	0.000000	177.54400	24.40760	0.726306	9	0
115262761	34.25500	2.446020	0.000000	110.80100	15.93850	0.718124	11	0

Figure 5: Dataframe met alle features gebruikt voor de decision trees

5 Methode

5.1 K-means

We tried different Machine Learning techniques to find the best method for this specific problem. As mentioned before the given rain tracks were established by the amount of tracks per minute (many tracks would probably indicate rain) and the proportion of bird flocks. This meant that the rain in the dataset was far from certain to actually be rain. For this reason we thought that this would maybe be a bit too inaccurate to use, so we decided to first use an unsupervised algorithm to find out if we could find some results by clustering the data. If we compared it to the given data and we would find some similarities then this would give a lot more insight in which data could be classified incorrectly. The K-means Algorithm is very useful to find out if this would give some results.

At first this method looked really promising because we found an accuracy of 92%. The problem here was that most of the data wasn’t rain and the algorithm classified most of the data as no rain. This meant that the algorithm could only classify ‘no rain’ accurately. We only found 0,2% of rain classified correctly(99,8% false negatives). This is a prime example of the incapability of similarity to accurately score an algorithm. It turned out that the K-means algorithm was not capable of dividing the data into rain and non-rain.

We think that this algorithm does not work because the rain is divided over all sorts of clusters within the data consisting of anything else than rain. We also know that there is other types of clutter, which means that it is not a two-class problem. However after extensively testing the algorithm with a variety for k-values, the accuracy didn’t improve. This makes it impossible to cluster groups unsupervised as rain and no rain.

5.2 Bayes Classifier

This meant we had to try a supervised algorithm. For a supervised algorithm it would be nice to know which features divided the data best in rain and no rain. With this extra information it was possible to try feature selection and it turned out that the features ‘score’ and ‘phi diff’ respectively were the feature that divided the data the best. The first thing we tried was the Bayes classifier.

For the Bayes Classifier we first needed to find the variables that divided the rain and non-rain tracks the best. To find these variables we used *Feature Selection*. Logistic regression gave us insight in the ranking for the Feature Selection. This algorithm takes into account the mean and standard deviation of each variable for the tracks labeled as rain and the tracks labeled as non-rain. The algorithm ranked seven variables from most similar (1) to most different (7). The variables which were most responsible for dividing the data turn out to be ‘score’ (7) and ‘rho diff’ (6). With these features we tried to build a Bayes Classifier. This worked for one variable. When we used two variables we could not classify the test set anymore because all the posteriors returned an infinite number and therefore the numbers couldn’t be compared.

To find out why the Bayes classifier performed so poorly for dividing the data into rain and no rain, we looked deeper into the features. With feature selection we could only find a ranking and not how well the feature performed when dividing the data. To get more insight we calculated the mean and the standard deviation of every feature and plotted for every feature the graphs of the probability for the values of that specific feature. We found out that all features quite poor at dividing the data into rain and no rain. Feature selection ranked ‘score’ as the best feature. We decided to plot the best feature (score) and the worst feature (rho diff). These plots are illustrated below.

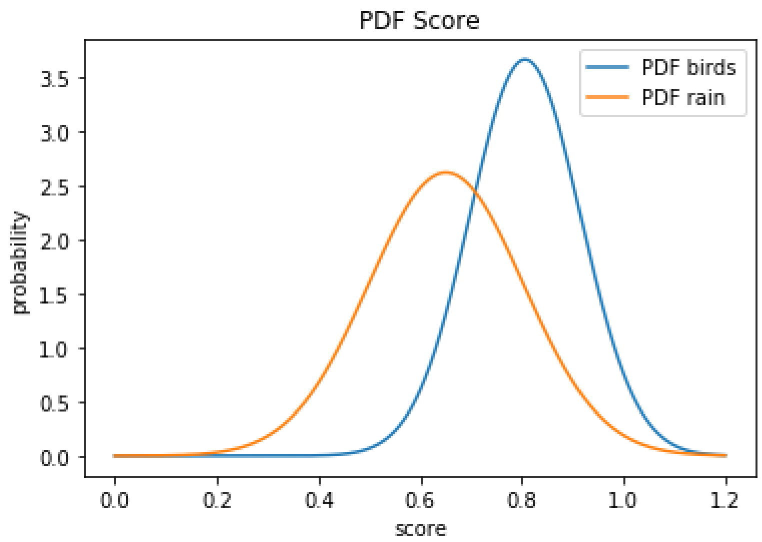

Figure 6: PDF Score

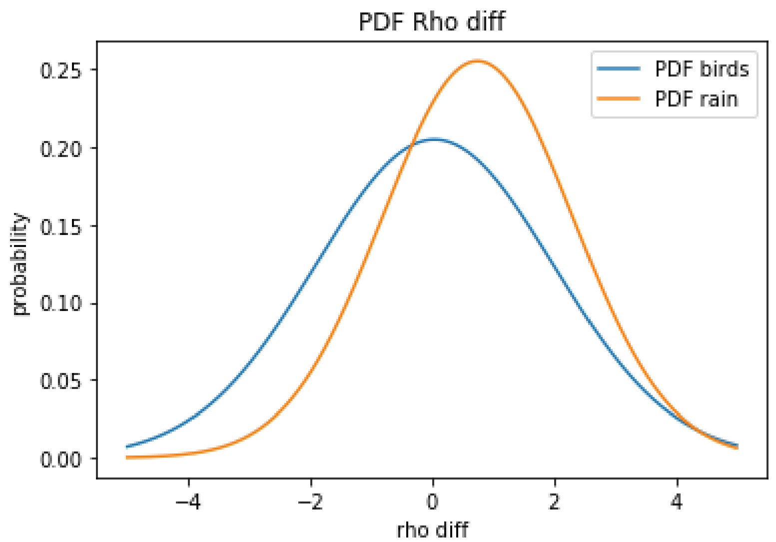

Figure 7: PDF Rho_diff

As visible in this plots, the density functions are overlapping severely, even when we look at the best features for dividing the data. This meant that this method also did not perform well.

5.3 Random Forest: Overview

To solve this problem we had to find a supervised method which works fine with a lot of different features. The decision trees method is a way to use every feature and was by far the best Machine Learning technique we tried. The benefit of this technique is that all variables are used to compare rain tracks with no rain tracks. Because of the decision to only use date from September, there was a risk of overfitting to that month if only one tree would be trained on the data. To minimize this factor and to improve classification accuracy, we implemented a random forest containing containing many random trees, rather than just a single decision tree (Deschamps et al., 2012, Ali et al., 2012, Banfield et al., 2006). For more details, read paragraphe 5.4.

5.4 Random Forest: Details

Each tree is fitted to a random subsample of the data, after which all the fitted trees are weighed in relation to each other (averaging) to form a classification model. Most of the model creation was done with the RandomForestClassifier, which is an ensemble classifier within the Scikit-learn library. An ensemble classifier, in general, combines a number of algorithms to perform better and have more robustness compared to using a single algorithm (Polikar, 2006). As mentioned before, a new feature was made that tells whether a track should contain rain or not. This new column was here used as a target for the model to train towards. All the data was split into a training and testing set with 70%, 30% size respectively. The model was trained using 10 trees, because this was the default number of trees that was used in our package. There were only very minor improvements (<0.1%) when more trees were used.

6 Results

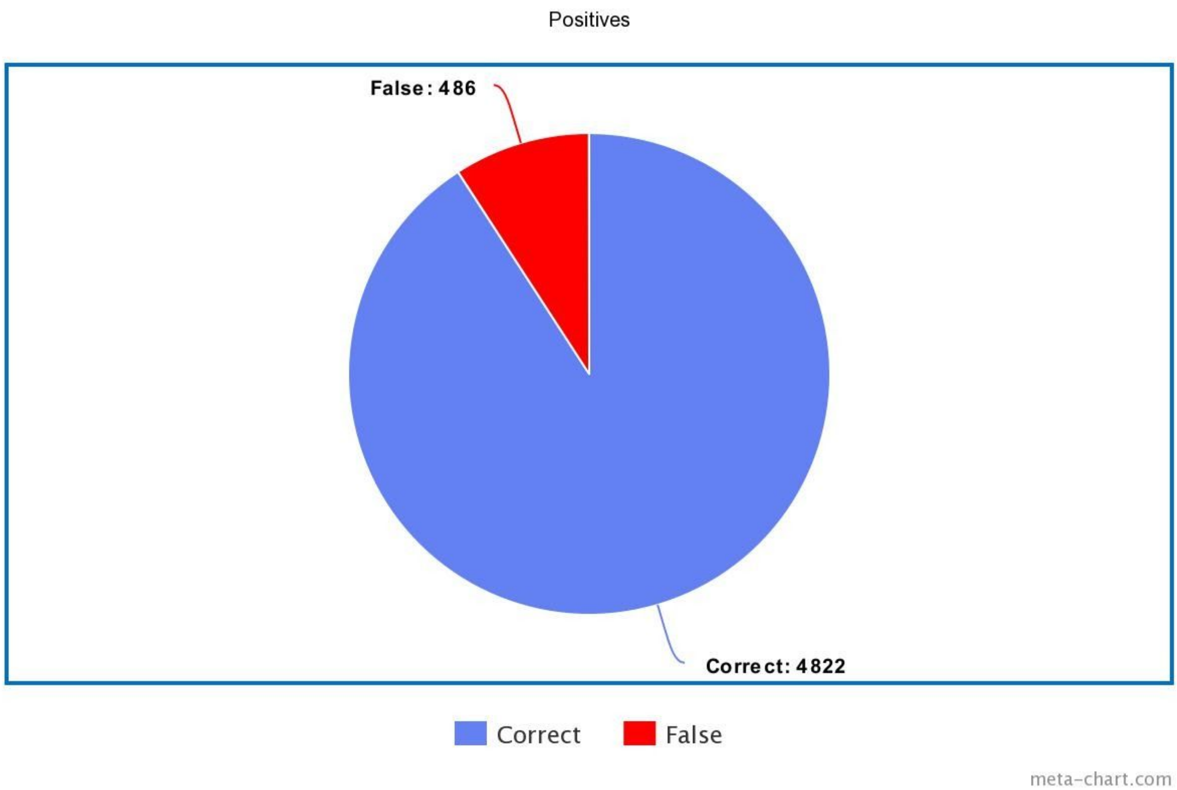

Figure 8 Rain: 9,2% false positives(red) and 90,8% correct positives(blue)

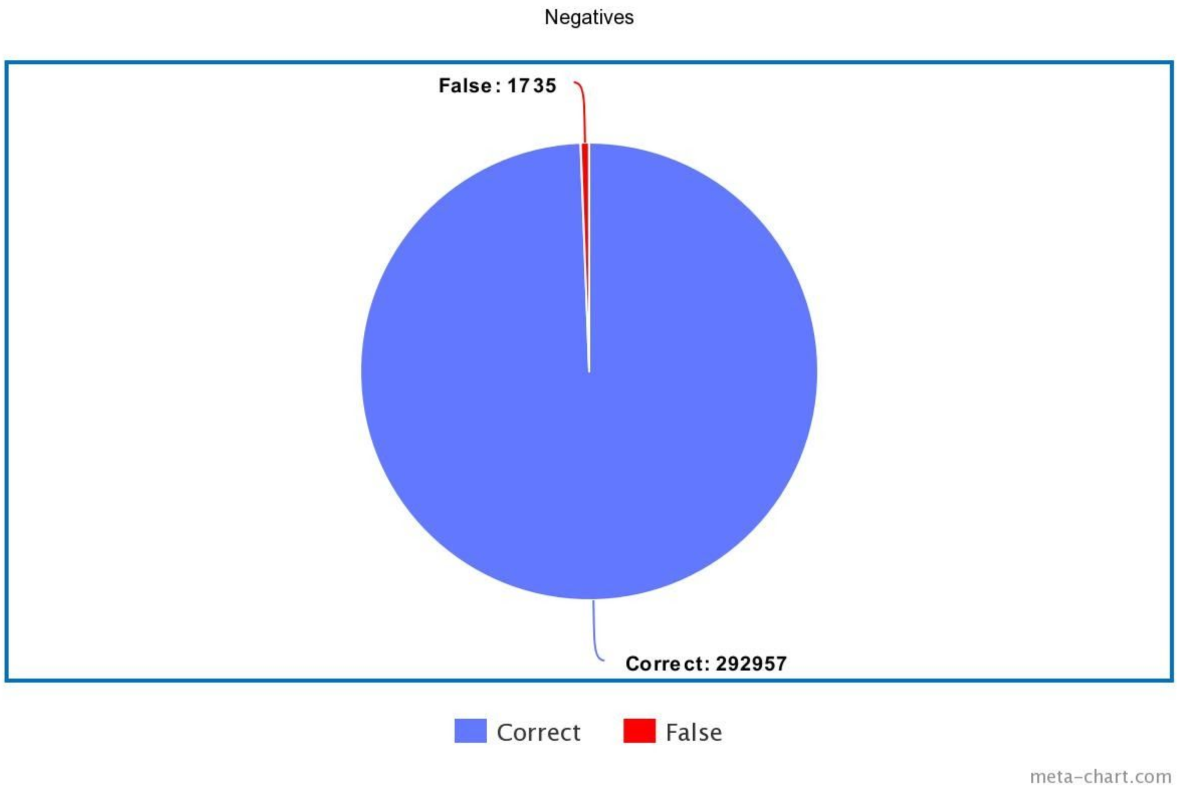

Figure 9 No Rain: 0,6% false negatives(red) and 99,4% correct negatives(blue)

When we trained the dataset with a ratio of 0.3 (test/training) we found that most data points in our test set were labeled correctly (99,3%). We found that about 90,8% of the data labeled as rain(positives) was labeled correctly based on the given rain id's(Figure 8). For the data labeled as no rain(negatives) we found that 99,4% was labeled correctly as no rain (Figure 9).

This means we only have 0,6% false negatives, meaning rain being classified as non-rain. This percentage could never be zero because of the inaccuracies in the training set. The amount of false negatives is minimized.

We also find 9,2% false positives, meaning non-rain being classified as rain. This result was also predicted because Jens asked us to extract rain data from the dataset that he might have missed. This meant that there were still rain tracks in the dataset that were just labeled as non-rain.

This 9,2% consists of new rain data that we found using the algorithm. We have collected the id's of these new rain tracks, and saved them in an allocated file.

7 Conclusion

In conclusion, it has become apparent that the usage of machine learning algorithms to classify tracks collected by the Robin Radar, can greatly improve the quality of the dataset. Our implementation of a random forest classifier was most successful in differentiating tracks with rain from tracks without rain. With an average accuracy score of 99.4% and over 9% extra rain tracks found, we can conclude that the random forest classifier is an immense improvement compared to the old, crude way of classifying rain.

8 Discussion

For the sake of transparency we did not use a neural network because this would create a ‘black box’ where it would be difficult, if not impossible, to explain why our model would work in the way it does. In our opinion it was necessary to have some insight in what happens inside the machine learning algorithm and to have visible steps.

We mostly used supervised learning techniques. To use a supervised method we had to make the big assumption that the training data is correct. Unfortunately, we didn’t know for 100% certain that all of the data used for training the data was accurate. Because of the strong imbalance between the classes this may have negatively affected our classification power. This also has a benefit, we now found many new rain tracks and no rain tracks which were misclassified before. These tracks are ambiguous and may be classified differently after further and more thorough research.

Most of the features were overlapping a lot because the rain was classified in all sorts of groups. This made it really hard to find a model that was able to separate the data into rain and no rain. We did not use the trajectories because we were unable to read them properly from the .csv file in way that was usable in Python and implements the values in a meaningful way. Maybe the trajectories could be a better feature to separate the data because the trajectories in a rain shower would probably be rotating randomly at every new measurement. This follows from the fact that every raindrop is connected randomly to another raindrop, which is then evaluated as a track.

8.1 Future work

To check if the rain id’s found by the algorithm are actually rain there are two ways to get a better insight if they are labeled correctly. First of all, it is possible to check if these id’s are within a certain time span near a lot of other tracks that are classified as rain. If this is the case, the probability that the next track is rain is much higher than if there would not be any rain measured around that time. Another way to check this is to compare it to the actual weather that day.

After classifying all these id’s it is possible to run the algorithm again. The new classified rain influences the outcome of the algorithm. This could, possibly, result in better scores and new rain tracks.

9 References

Ali, J., Khan, R., Ahmad, N., & Maqsood, I. (2012). Random forests and decision trees. International Journal of Computer Science Issues (IJCSI), 9(5), 272.

Banfield, R. E., Hall, L. O., Bowyer, K. W., & Kegelmeyer, W. P. (2006). A comparison of decision tree ensemble creation techniques. IEEE transactions on pattern analysis and machine intelligence, 29(1), 173-180.

Deschamps, B., McNairn, H., Shang, J., & Jiao, X. (2012). Towards operational radar-only crop type classification: comparison of a traditional decision tree with a random forest classifier. Canadian Journal of Remote Sensing, 38(1), 60-68.

Polikar, R. (2006). Ensemble based systems in decision making. IEEE Circuits and systems magazine, 6(3), 21-45.