Report Project 3

Erlend Øydvin
(Dated: December 17, 2021)

Abstract

In this project measured reflectivity from a weather radar has been compared with measured precipitation from a rain gauge in the same area. The main goal was to use a Feedforeward Neural Network and Logistic regression to identify wet and dry periods based on measurements from the weather radar. The results shows that the Neural Network performs better than Logistic Regression and that the neural network is able to give a hint about whether it is a rainy period or not. None of the methods was able to, for the accumulated time interval (7,5 min or 1 hour), correctly classify a rainy period, often missing the rain by a couple of hours. Thus weather radar still seems to only give an indication for the probability of rainfall.

I. INTRODUCTION

In this project we will study reflectivity from a weather radar image and compare it with what is measured on the ground in a rain gauge. The goal is to train an algorithm to identify wet and dry periods. Determining if its raining or not can be useful for quality control and filtering of other precipitation data sources such as commercial microwave links (CML) and personal weather stations (PWS) [2, 4]. In this report we study the performance of a simple Feedforward neural network and Logistic regression to analyse weather radar measurements and predict whether its raining or not. We then study how accumulating rain and radar measurements over longer period affects the predictions. The first part explains the theory and methods used, the second part shows and discusses the results.

Weather radar data was downloaded from Norwegian Meteorological Institute's OPENDAP service Thredds. The python library netCDF4 was used to download a 4x5 km grid over Ås municipality. The rain gauge data was downloaded using Norwegian Meteorological Institute's API Frost and the python package requests. The study area with radar grid and rain gauge can be seen in Figure 1. The period used in this dataset was from 1th of July 2018 to 11th of September 2018, where the first 75 percent of the days was used for training and the last 25 percent of the days was used for validation. The pyhton code can be found at the following git-repository: `https://github.com/erlnwind/FYS-STK4155/tree/main/Project_3`

II. THEORY

Measuring precipitation

A weather radar works by sending out an electromagnetic pulse. Reflectively is the amount of transmitted power returned to the radar receiver after hitting precipitation. The more precipitation, the more power is reflected back. Reflected power can thus be related to rain rate. In contrast to point observations, such as rain gauge, weather radar has the potential to cover large areas. However, despite advances in radar technology, beam blockage and the vertical profile of reflectivity still remains a limiting factor [2]. Due to these limiting factors it can be hard to relate the precipitation measured by the radar to the precipitation that is measured on the ground. Conventional precipitation maps are thus created by calibrating measured radar reflectivity with rain gauges on the ground [1].

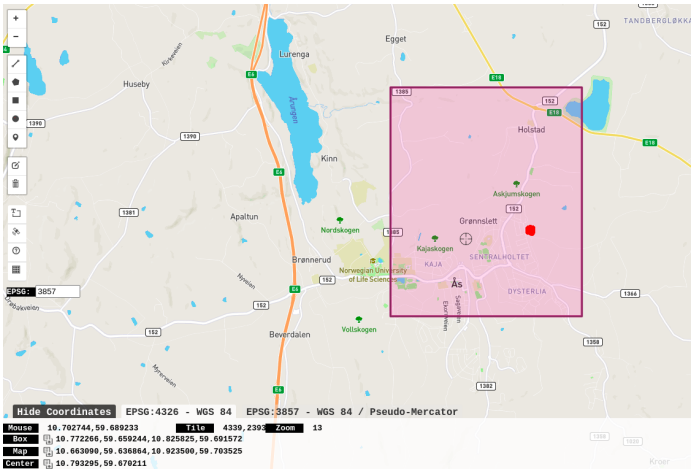

FIG. 1: 4x5km radar area in Ås municipality (red square). Rustad weather station is indicated with a red dot.

Classification of weather events

Weather events can be classified into different categories based on for instance raindrop size, intensity, spatial and temporal structure and wind. Maybe the most important category is whether it is raining or not [3]. In this project we use weather radar images and compare them with measurements on the ground. The goal is not to predict how much it is raining, but to classify rainy periods from not rainy periods. The classification is done using Logistic regression and a Feedforeward neural network.

Logistic Regression

The main goal for the classification problem is to minimize the cross entropy given by

$$\mathcal{C}(\vec{\beta}) = -\sum_{i=1}^{n} y_i \ln[P(y_i = 0)] + (1 - y_i) \ln[1 - P(y_i = 0)] + \lambda \frac{1}{2}\beta^2, \tag{1}$$

where the logistic function is defined by

$$P(y_i = 1|x_i, \hat{\beta}) = \frac{\exp\{(\beta_0 + \beta_1 x_i)\}}{1 + \exp\{(\beta_0 + \beta_1 x_i)\}}, \tag{2}$$

The main goal in logistic regression is to minimize Equation 1 with the logistic function defined by Equation 2. By taking the derivative of Equation 1 with respect to its parameters β yields

$$\frac{\partial \mathcal{C}(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = -\boldsymbol{X}^T (\boldsymbol{y} - \boldsymbol{p}) + \lambda \boldsymbol{\beta} \tag{3}$$

This expression can be used to minimize the cost function by using for instance an gradient descent algorithm.

Neural network

The first step in a FFNN is to compute the weighted sum of the input features for each neuron in the input layer as follows

$$z_j^l = \sum_{i=1}^{F} w_{ij}^l x_i + b_j^l. \tag{4}$$

Then each input z_j is passed through a activation function

$$a_j^l = f(z_j^l). \tag{5}$$

Then this process is repeated for every layer l in the network up to the output layer denoted by L. In the output layer z_j^L is calculated for each neuron and then passed to the output function. For the regression problem the output function is given by the linear function

$$a_j^L = f(z_j^L) = z_j^L, \tag{6}$$

while for the classification problem the output function is given by the softmax function

$$a_j^L = \frac{\exp\{(z_j^L)\}}{\sum_{c=0}^{C-1} \exp\{(z_c^L)\}} \tag{7}$$

After doing doing the feed forward we compute the predicted error and update the weights accordingly. This is done with the back-propagation algorithm. The first step in back-propagation is to calculate

$$\delta_j^L = f'(z_j^L) \frac{\partial \mathcal{C}}{\partial (a_j^L)} \tag{8}$$

for the output layer where the cost function C is given by Equation 13 and $f'(z_j^L)$ is the derivative of the activation function. Then the next steps is to compute

$$\delta_j^l = \sum_k \delta_k^{l+1} w_{kj}^{l+1} f'(z_j^l) \tag{9}$$

for all layers back to the input layer. Back-propagation constructs new weights and we can update the weigths and biases as follows

$$\begin{aligned} w_{jk}^l &\leftarrow= w_{jk}^l - \eta \delta_j^l a_k^{l-1}, \\ b_j^l &\leftarrow b_j^l - \eta \frac{\partial \mathcal{C}}{\partial b_j^l} = b_j^l - \eta \delta_j^l, \end{aligned} \tag{10}$$

III. METHOD

Classifying rain and no rain

The rain gauge data from Rustad weather station has a temporal resolution of 1 minute. To compare it with the weather radar an accumulated sum for 7,5 min and 1 hour was computed. The rain was then converted to binary rain and no-rain by saying that it was rainy if it recorded any precipitation above 0 mm/h and no rainy otherwise. The weather radar on the other hand measures reflected power for 1kmx1km grids every 7,5 minutes. In this report we have studied a 4x5 km grid. The radar reflectivity for 1h was computed using an accumulated sum of every measurements within that hour. Then for the Logistic regression method each radar cell was given a weight ω_i, added together with the other cells and then compared with what was measured in the rain gauge. For the neural network each radar cell was fed into the input layer, and then compared with the rain gauge.

Error measurement

The performance of the neural network and logistic regression was measured using the following definition of accuracy score

$$accuracy = \frac{\sum_{i=1}^{n-1} I\left(t_i = y_i\right)}{n} \tag{11}$$

where t_i is the predicted classification and y_i is the correct classification. An interesting thing about rain is that it is most often not raining, and predicting no rain is often a very good guess. Thus the validation predictions was compared with the no rain prediction.

Hidden layers and activation functions

In the logistic regression we can experiment with the learning rate and the regularization term. In the neural network we can in addition experiment with the choice of activation function, number of epochs and the architecture of the neural network. In this project I have experimented with the sigmoid activation function given by

$$f(x) = f(x) = \frac{1}{1 + e^{-x}} \tag{12}$$

and the rectified linear unit (RELU) function

$$ReLU(x) = \begin{cases} 0 & z < 0, \\ z & z \geq 0. \end{cases} \tag{13}$$

train/test and K-fold cross-validation

In machine learning it is common to split the dataset into a train group and a test group. The algorithm is first trained against the train group and then tested against the test group. This routine can expose over-fitting of data,

a case where the model gains accuracy by modeling noise. By testing it against test data the the error will be higher than in the train data, indicating that the model is over fit. k-fold cross validation works by shuffling the data-set and dividing it into into k groups. Then one group is taken out as test group and the remaining groups are used to train the algorithm. This is repeated for all groups. The resulting accuracy score is then obtained by taking the mean accuracy score of all groups. k-fold cross validation us good as we use more of the data and we can be more confident that the scores we get are correct as we do k tests with different chunks from the dataset.

IV. RESULTS

Train and test grid search

Optimal parameters for the neural network and logistic regression method was found using k-fold cross-validation on data from 1st of July 218 and to 23rd of August 2018. The MLPClassifier from scikit-learn was used. Different network architectures was tested and for each network architecture and activation function a grid search with different learning rates (0,1 to 0,001) and regularization parameters (0 to 0,4) was used. Table I and Table II summarizes the best accuracy scores for different network architectures and the corresponding learning rate and regularization parameter. We can observe that the accuracy scores generally are very high for both the neural network and the logistic regression and that increasing the neural network complexity decreases the accuracy scores. Interestingly guessing zero rain also produces a good accuracy score. The ReLU activation function seems to generate slightly better results than the logistic activation function. Increasing or decreasing the Logistic Regresssion regularization parameter did not seem to prevent any kind of over-fitting and thus only one regularization parameter has been listed in the report. When comparing accuracy scores for the 7,5 min and 1 hour accumulation time we can see that the accuracy score is generally higher for the 7,5 minute than for the 1 hour accumulation time.

Prediction on validation dataset

The validation period was set from 24th of August 2018 to 10th of September 2018. The neural network was now trained using the optimal parameters and data from the whole train/test period (1st of July 218 - to 23rd of August 2018). Interesting Accuracy scores for the validation dataset are listed in Table III and Table IV. Like in the train and test datasets we get generally very high accuracy scores and guessing zero rain also produces a high accuracy score. Increasing the number of neurons in the neural network decreases the accuracy score. In Figure 2 i have plotted measured rainfall for Rustad weather station and identified wet periods based on the radar images. To make it more readable the plot does not cover the whole validation period but only values from the 5th of September. In the plot we can see that Logistic regression is not able to identify rainy periods for 7,5 min accumulation period. For the 1 hour accumulation period Logistic regression is able to identify one of about nine rain events. We can also observe that the neural network with a small number of neurons also does a poor job of detecting rain periods. Increasing the number of neurons causes the neural network to more correctly identify rainy periods. Further, the neural network was able to identify more wet periods using 7,5 minute accumulated precipitation compared with 1 hour accumulation time. The identified wet periods are not always exactly in the correct spot, sometimes missing the rain event by a few hours.

	Accuracy score	Learning rate	Regularization
NN Logistic [4, 4]	0,98	0,001	0,0
NN Logistic [8, 8]	0,97	0,010	0,0
NN Logistic [15, 15]	0,97	0,001	0,0
NN Logistic [30, 30]	0,96	0,010	0,0
NN Logistic [40, 40]	0,96	0,001	0,0
NN Logistic [50, 50]	0,96	0,001	0,0
NN ReLU [4, 4]	0,98	0,001	0,2
NN ReLU [8, 8]	0,98	0,010	0,4
NN ReLU [15, 15]	0,97	0,100	0,0
NN ReLU [30, 30]	0,97	0,100	0,0
NN ReLU [40, 40]	0,97	0,010	0,3
NN ReLU [50, 50]	0,97	0,010	0,1
Zero rain guess	0,97	-	-
Logistic Regression	0,98	-	10

TABLE I: k-fold cross-validation accuracy score for 7,5min accumulated precipitation for the train/test dataset. The [m, n] symbolize m and n numbers of neurons in the two hidden layers.

	Accuracy score	Learning rate	Regularization
NN Logistic [4, 4]	0,93	0,010	0,0
NN Logistic [8, 8]	0,93	0,010	0,0
NN Logistic [15, 15]	0,92	0,100	0,0
NN Logistic [30, 30]	0,90	0,010	0,3
NN Logistic [40, 40]	0,90	0,010	0,0
NN Logistic [50, 50]	0,91	0,010	0,0
NN ReLU [4, 4]	0,94	0,001	0,0
NN ReLU [8, 8]	0,94	0,001	0,2
NN ReLU [15, 15]	0,93	0,001	0,4
NN ReLU [30, 30]	0,92	0,100	0,1
NN ReLU [40, 40]	0,92	0,010	0,1
NN ReLU [50, 50]	0,92	0,100	0,0
Zero rain guess	0,94	-	-
Logistic Regression	0,94	-	10

TABLE II: k-fold cross-validation accuracy score for 1 hour accumulated precipitation for the train/test. The [m, n] symbolize m and n number of neurons in the two hidden layers.

	Accuracy score	Learning rate	Regularization
NN Logistic [4, 4]	0,95	0,001	0,0
NN Logistic [40, 40]	0,92	0,010	0,4
NN ReLu [4, 4]	0,95	0,001	0,0
Zero rain guess	0,95	-	-
Logistic Regression	0,95	-	10

TABLE III: Accuracy score for 7,5 minute accumulated precipitation on validation dataset. The [m, n] symbolize m and n number of neurons in the two hidden layers.

	Accuracy score	Learning rate	Regularization
NN Logistic [4, 4]	0,87	0,010	0,0
NN Logistic [50, 50]	0,87	0,010	0,0
Zero rain guess	0,88	-	-
Logistic Regression	0,89	-	10

TABLE IV: Accuracy score for 1 hour accumulated precipitation on validation dataset. The [m, n] symbolize m and n number of neurons in the two hidden layers.

V. DISCUSSION

The results shows that the accuracy score is best for Logistic Regression and neural networks with few neurons. This could indicate that increasing the number of neurons causes over-fitting of the data. However, by visual inspection on the validation dataset it seems that these simple models gives a good fit because they lean towards guessing zero rainfall, only guessing rain when it is very clear that its raining. As guessing zero rain also gives a very good accuracy score this could indicate that guessing rain is risky, lowering the accuracy score most of the time. Although the results shows that the accuracy score decreases for more complicated neural network architectures, the visual results shows that these more complicated models is able to identify rainy periods, but not necessarily the exact moment it is raining. This could indicate that the model is able to guess periods with high probability of rainfall, but not the exact moment the rainfall occurs. Since guessing zero-rain is such a good guess then the model will not gain accuracy by trying to guess rain in rainy periods. The reason for the low accuracy score thus could come from how I have defined a good guess. By redefining the accuracy score to give a correct classification if the guess is within a time interval, rather than a correct classification the exact moment, the accuracy score for Logistic regression and the neural network could and up behaving more like the visual inspections on the validation dataset.

Accumulating reflectivity and rain gauge data over 1 hour seem to make it easier for Logistic regression to identify rainy periods. This could be due to that this causes the rainy period to be more consistently defined in time, not jumping in and out every 7,5 minute making it easier for the regression algorithm to find a direct relationship between radar and measured rain. The neural network on the other side seems to do a slightly better job when studying 7,5 minutes accumulated rainfall, see for instance the precipitation event around the 6th of September where the 7,5 minute trained neural network seems to be able to predict a longer precipitation event than the one trained on 1 hour accumulated values. This could indicate that that the neural network is able to model some more complicated physical structure between measured weather radar reflectivity and measured rainfall on the ground.

VI. CONCLUSION

Measured radar reflectivity was used with a neural network and logistic regression to predict wet and dry periods on Rustad weather station. The neural network did a better job than the logistic regression, especially for architectures with many neurons. The accuracy score could in further work take into consideration that radar reflectivity gives a probability of rain rather than measuring rain directly. This could be done by predicting longer intervals of rain rather than the 7,5 minute or 1 hour accumulation. As this an analysis of time series and images further work could investigate a convolutional and/or a recurrent neural network. Another interesting aspect to investigate could be to increase the size of the weather radar grid, if accuracy is increased this way it could indicate an relationship in the spatial structure of measured reflectivity.

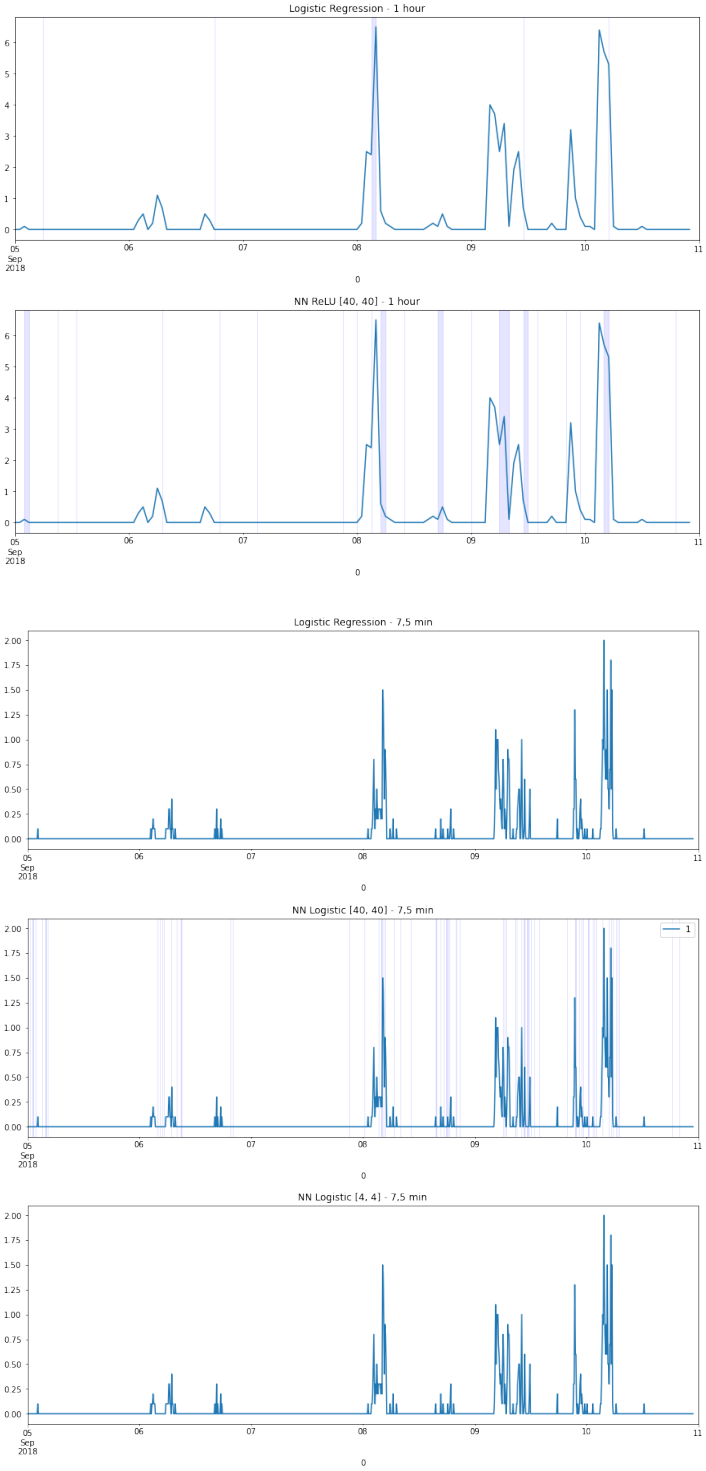

FIG. 2: Predictions on validation dataset for different accumulation periods, neural network architectures and logistic regression. The blue line is measured precipitation (mm/7,5 min or mm/h) from Rustad weather station. The vertical blue lines are identified wet periods based on measured weather radar reflectivity.

REFERENCES

[1] Berne, A. and Krajewski, W. (2013). Radar for hydrology: Unfulfilled promise or unrecognized potential? *Advances in Water Resources*, 51:357–366.

[2] Chwala, C. and Kunstmann, H. (2019). Commercial microwave link networks for rainfall observation: Assessment of the current status and future challenges. *WIREs Water*, 6(2).

[3] Leblois, E. and Creutin, J.-D. (2013). Space-time simulation of intermittent rainfall with prescribed advection field: Adaptation of the turning band method. *Water Resources Research*, 49(6):3375–3387.

[4] Messer, H. and Sendik, O. (2015). A new approach to precipitation monitoring: A critical survey of existing technologies and challenges. *IEEE Signal Processing Magazine*, 32(3):110–122.