CS7CS4 - Machine Learning - Group 90

Predicting Wave Height from the Irish Weather Buoy Network Data

Conor Dalton - Leo Cooke

May 30, 2023

1 Introduction

The Irish marine institute provides access to open source weather data from a number of its off-shore buoys around the Irish coast. We thought it would be interesting if we could use a combination of the many measurements these buoys take to predict another weather feature. These readings happen every hour and there are years' worth of data available. Wind speed, atmospheric pressure, sea temperature and wave height were all features we thought we could use to create a prediction model. We decided to investigate predicting wave height using the other readings as we felt that this feature was the most influenced by local conditions around the buoy. The inputs to our models are wind speed km/h and atmospheric pressure kPa with wave height m the target prediction. We used a variety of different models on the data including decision trees, ridge and lasso regression and Multi-layer Perceptron regressor (MLP). Time-invariant and future wave height predictions were analysed. We decided to use buoy *M2* in figure 1 in the data which is situated off the east coast of Ireland as this provided the most consistent data over the longest period.

Figure 1: Location of M2 buoy

2 Data set and Features

As stated, the dataset we used was the data collected from the M2 weather buoy in the Irish sea collected since 2001. This data is available online and we based our data collection code on previous work[1] which we made use of and stripped down, as many of the readings were not beneficial to us. Data is collected (almost) every hour. For the M2 buoy there are 151,906 data entries. For our models we used atmospheric pressure, wind speed and wave height so it was important that we only used data entries that contained data entries for these quantities. We removed any data entry where any of these data were missing. This reduced the size of the dataset to 118,330 data points. This data was appropriate for our time-invariant experiments, we used the atmospheric pressure and wind speed data to determine the wave height at that moment in time. Only the data for the multi-layer perceptron was normalised between 0 and 1 as this is important for training neural nets. This speeds up learning and leads to faster convergence.

```
array([[   7.98 , 1014.8  ],
       [   7.98 , 1014.2  ],
       [   8.95 , 1014.2  ],
       ...,
       [  18.358, 1029.028],
       [  16.508, 1029.688],
       [  15.512, 1029.98 ]])
```

```
0         1.100
1         1.000
2         0.800
3         0.800
4         0.900
          ...
118324    1.875
118325    1.563
118326    1.719
118327    1.563
118328    1.406
```

Figure 2: Example of processed data with inputs on the left and targets on the right

Data in the format in figure 3 is not appropriate for our time-dependent experiments as they

require consecutive data points. To fix this we removed any data on a day without 24 data entries. This again reduced the number of data points to 80,137.

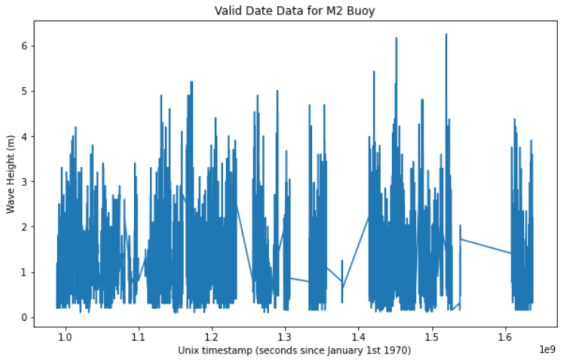

Figure 3: Data from buoy with gaps

This data was arranged into blocks of 24 hours, 12, hours, 6 hours, 4 hours, 3 hours and 2 hours. These blocks of data used a number of the features from the earlier hours as input features and the final wave height data point as the target feature.

	wind1	wave1	atmo1	wind2	wave2	atmo2
0	2.920	0.600	1020.000	4.090	0.400	1020.800
1	4.090	0.400	1020.800	1.950	0.400	1021.200
2	2.920	0.400	1021.400	4.090	0.400	1021.800
3	4.090	0.300	1022.400	4.090	0.300	1023.000
4	4.090	0.300	1023.800	6.030	0.300	1024.000

Figure 4: Example of consistent data

The data in Figure 4 seemed more appropriate than having a rolling window move over a set of consecutive data points as there are so many times where data was unavailable and interpolation for such chaotic data will be very inaccurate.

3 Methods

We used a variety of different learning algorithms on our data and analysed their performance to see which model gave us the most accurate predictions. Each works in a different way so it was important to try a wide selection to see what best fit our data. We used the sci-kit learn package[2] in python to implement our chosen algorithms. We began by modelling time-invariant regression on the data and then moved on to predicting future wave heights using previous data records.

Initially we used linear models, ridge regression and lasso regression. Ridge regression performs L2 regularization which shrinks the coefficient estimates towards 0. L2 regularization uses the cost function in (1).

$$Loss = Error(Y, \hat{Y}) + \lambda \sum_{1}^{n} w_i^2 \qquad (1)$$

λ is the penalty term where higher values of alpha produce a bigger penalty and therefore a greater reduction in the magnitudes of the coefficients. Lasso regression is similar to ridge regression but uses L1 regularisation instead with the cost function shown in (2).

$$Loss = Error(Y, \hat{Y}) + \lambda \sum_{1}^{n} |w_i| \qquad (2)$$

Here the absolute magnitude of the coefficients is used in the penalty term of the cost function instead of the square. The difference between this and the previous ridge regression model is that lasso shrinks the less important feature's coefficient to zero which can remove some features altogether. This can be useful if a large number of features is used.

A decision tree model was also used which differs from the linear models used before. Decision tree regression uses a tree structure, breaking the data down into smaller and smaller subsets with increasingly homogeneous data in each subset. The standard deviation of a subset is used to group data together. This standard deviation is algorithmically reduced until the model can make accurate predictions from associated subsections of the data. An advantage of decision tree models is that missing data does not affect the construction of the decision tree, however, training decision tree models is computationally expensive and can take a long time.

Following these machine learning methods, we investigated using neural nets to more accurately predict our target values. We used a Multi-layer Perceptron (MLP) model which uses multiple layers of neurons with associated weights to make predictions. These weights are tuned using gradient descent to minimise a square cost function. More design choices need to be made when training a MLP model than the previous models with layer architecture, data normalisation, penalty term alpha, solver, etc. all to be optimized.

For the future wave height prediction model, ridge and lasso regression were again used. A lot more work on data preprocessing was needed for this task as past data and future training targets needed to be consistent in their time difference. The two input features were augmented into polynomials of higher degrees for all meth-

ods we used to increase the number of input features used in training.

4 Experiments/Results/ Discussion

4.1 Time-Invariant Regression

Ridge and lasso regression
The ridge and lasso linear regressors were both used in the same way to see which was more appropriate for this type of data. Using the atmospheric pressure and wind speed as input data to determine the wave height at a particular time. the ridge regressor predictions can be seen in figure 5

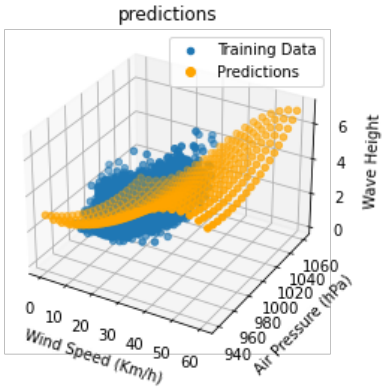

Figure 5: Training data and ridge regression predictions

The data was optimised by using every polynomial combination of the 2 features up to a power of 3 were used. Using higher values negatively affected the training time of the lasso regressor to the point where it became impossible to use.

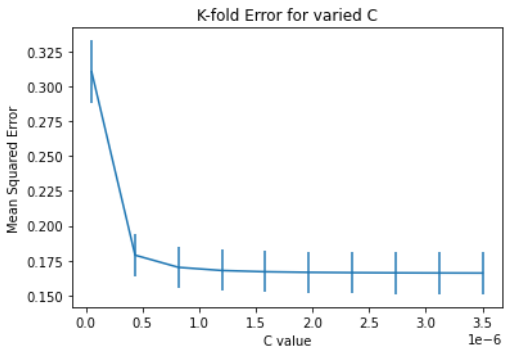

Figure 6: Cross validation for ridge regression

For both of these regressors, 5-fold cross-validation was used to determine the optimal value of alpha that was used - alpha determines. This is seen in figure 6. These values were very different from one another, (we chose a value of 0.00000005 for the ridge regressor and a value of 0.5 for the lasso regressor).

Decision tree
The decision tree regressor was used on just the 2 original input features. It was added more as an afterthought and the drastic difference between the training mean-squared-error and the validation mean-squared-error suggested that it was over-fitting as can be seen in table [NEEDREF] . Rather confusingly the predictions looked like they may have been reasonably accurate as can be seen in figure 7

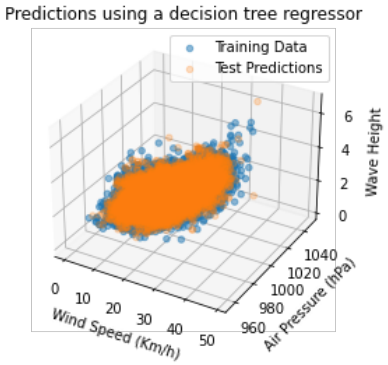

Figure 7: Training data and decision tree predictions

MLP
The MLP regressor model in sci-kit learn has a built in score method which returns the coefficient of determination of the prediction. The coefficient of determination is the proportion of the variation in the dependent variable that is predictable from the independent variable. This was used as a metric when analysing variations of the model. Mean-squared-error was also used as a metric with a value as close to 0 being optimal.

Initially, the model architecture had to be chosen. A MLP uses a multi-layered neural net with an input layer, an output layer and at least one hidden layer. As the two input features, wind speed and atmospheric pressure, were augmented to increase the number of features, multiple neurons were required in each layer. We chose four hidden layers with 16 neurons each. This number was chosen using general rules such as the number of hidden neurons should be 3/4 the size of the input layer, plus the size of the output layer. After experimenting with some different sizes, this was chosen for the final model.

The input data was normalised to lie between 0-1 before training as unscaled data can lead to an unstable or slow learning process in neural

networks. This data was then augmented with polynomials to increase the number of input features. A degree of 5 was used, although varying this value had little effect on the results. The hyperparameters of the model then needed to be chosen. The solver was specified as *Adam* which was sufficient for this training as other solvers such as *lbfgs* vastly increased training time with no model improvement. The *Adam* solver tunes the model parameters using stochastic gradient descent.

Alpha, which was the L2 regularization penalty term used, was then chosen using k-fold cross-validation. 5 folds were used which gives an 80%-20% split on training and validation data. A range of values from 0-1 was used and the value which gave the best coefficient of determination was chosen as the final model. This can be seen in figure 8.

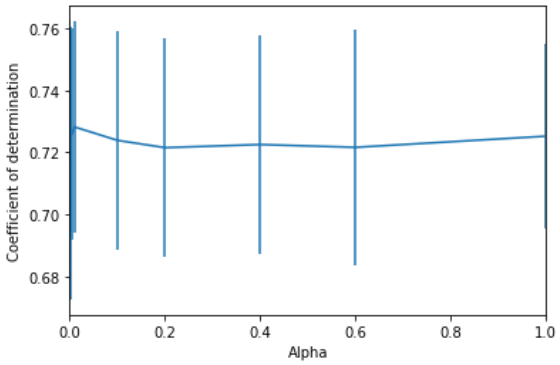

Figure 8: Cross validation for MLP

A value of 0.00005 was chosen for Alpha after cross-validation. The mean squared error during cross-validation was also calculated. This was useful when comparing the prediction mean squared error to ensure over-fitting of the data had not occurred. After choosing the hyperparameters and performing the data preprocessing, the final model was chosen.

This model returned a coefficient of determination of 0.74. This model also produced a mean squared error of 0.14 in the test data. Comparing this model's performance to a baseline model which gave a validation mean squared error of 0.22, it is clear that this model improves prediction performance.

4.2 Future Wave Height Prediction

The main task involved in the prediction was the selection and presentation of data described earlier in this report. The machine learning method used was, again, ridge regression. This choice was trivial.

Cross validation showed that altering the value of alpha seemed to have almost no effect until the value of alpha became quite high (over 5) where the mean-squared-error in prediction started to rise. The value used was 0.1 and this was kept constant throughout.

The data was arranged into blocks of consecutive-hour data points ranging from 24 to 2 hours. The target feature was always the final entry for wave height and the number of previous hour data points to take into account was varied.

The performance metric used was mean squared error as this gives information about how far off a particular prediction is. The baseline that was used for comparison was one that simply copied the value of wave height a certain number of hours earlier - corresponding to the number of hours of features that were being used by the ridge regressor. If the model was being used to predict wave height 2 hours into the future then the baseline model was the wave height 2 hours before the target.

For each block size predictions were produced and the mean-squared-error was calculated. A plot of the mean-squared-error with increasing time into the future can be seen in figure 9 where it is clear it increases linearly. We believe the main reason for this is because the features that you are using are less related to a value farther into the future.

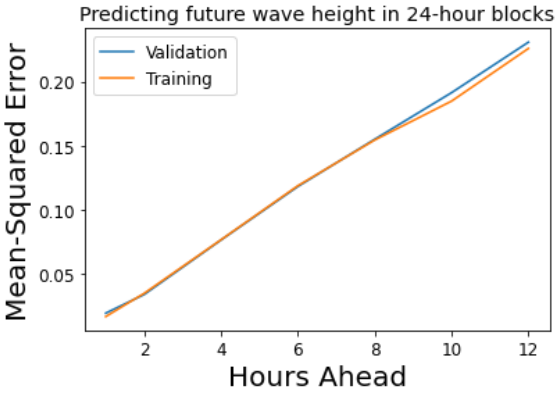

Figure 9: Error in future predictions

The improvement in performance compared to a baseline model of taking the mean wave height can be seen in figure 10.

Figure 11 shows the predictions for 1-hour ahead and 12-hour ahead prediction superimposed on the true values for wave heights. It can clearly be seen that the 1-hour ahead predictions very closely follow the true values. The-12 hour ahead predictions are not as close as the 1-hour ahead predictions but still follow the same patterns which is impressive.

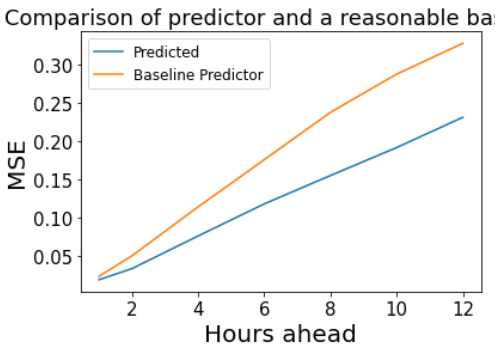

Figure 10: Performance compared to a baseline model

The regression models for predicting in the future reliably outperformed the baseline model described in experiments. This leads us to believe that this is an effective method of estimating wave height a few hours into the future.

5 Summary

Using a variety of different models, we had varying success in predicting our target variable, wave height, from our input features wind speed and atmospheric pressure. A table of our time-invariant models and their associated mean squared error can be seen below.

All of these models performed better than a baseline model of predicting the mean value of wave height with MPL performing the best but only marginally. The use of a neural net and increased training complexity probably gave it a very slight advantage

We were also successful in predicting future wave heights with linear models which produce meaningful predictions which could be useful to weather forecasters or surfers for example. The accuracy of predictions decreased the further into the future we tried to predict as expected and shows the unstable nature of weather data.

Overall we successfully extracted and processed data, analysed various models in predicting a target variable and were successful in creating relatively accurate prediction models for both time-invariant predictions and future wave height predictions. A workflow of cross-validation and iterative improvements was followed which produced the best results.

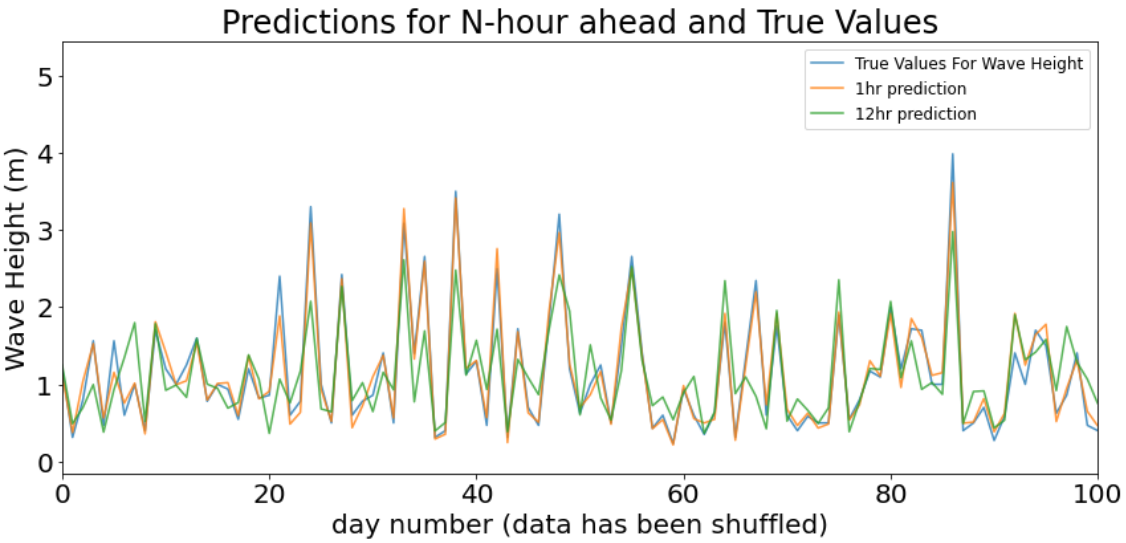

Figure 11: Predictions and true values

Model	Mean-squared-error (training)	Mean-squared-error (test)
Ridge regression	0.139549	0.144564
Lasso regresssion	0.13988	0.144817
Decision tree	0.059771607	0.221001298
Multi-layer Perceptron	0.140769355	0.142038614
Dummy predictor	0.540886	0.540886

Table 1: Mean-squared-error for each time-invariant regression technique.

6 Contributions

- The initial research was done by Leo Cooke.
- The code used to get the original data was adapted from Andrew Conway's code[1] by Conor Dalton.
- The data parsing, sorting and cleaning code was written by Conor Dalton.
- The Linear Regressor and Decision Tree code was written by Conor Dalton and based upon code submitted previously and code started by Leo.
- The Multi-Layer Perceptron Regressor Code was written by Leo Cooke.
- The future prediction code was written by Conor Dalton.
- The majority of the report was written by Leo Cooke.

Code is available at https://github.com/daltHub/machine_learning_project
LC
CD

References

[1] Andrew Conway. Ogi mi pilot. https://github.com/IrishMarineInstitute/OGI_MIPilot, 2019.

[2] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.