EMATM0044 Introduction to AI Coursework Part 1

Introduction

The task dictates that the ratio variable, net hourly energy usage of the plant (PE), be predicted using four other ratio variables, Temperature (T), Ambient Pressure (AP), Relative Humidity (RH), and Exhaust Vacuum (V), which represent hourly averages for the plant. As the output variable, PE, is continuous, the problem requires a regression algorithm that maps features to the 1-dimensional real number space ($\mathbb{R}$) (Russell & Norvig, 2021, p. 696). Furthermore, a supervised learning algorithm is needed to learn a function from features to output labels (Russell & Norvig, 2021, p. 695).

Methods

Data Pre-processing

Pre-processing is an important step to undertake before modelling. The dataset was queried for any missing data and outliers that may skew model predictions. It was discovered that there were no missing data, and minimum and maximum values for each of the five variables reside within 3.5 standard deviations of their respective means, implying no material outliers.

Baseline Model

Initially, each input variable was graphed against the target variable to explore any relationships that may be insightful for modelling the data. It can be seen immediately that the variable, AT, has a strong negative correlation with the target variable (see Fig. 1). This linear relationship can be modelled effectively using a univariate linear regression model. Linear regression models the output variable as a linear combination of the input variables plus some constant and error term using the equation:

$$y = \beta_0 + \beta_1 x_1 + \cdots + \beta_n x_n + \epsilon.$$

Univariate linear regression is amongst the least complex supervised learning algorithms for mapping data to $\mathbb{R}$. It is computationally inexpensive and provides a useful baseline for comparing more complex methods against.

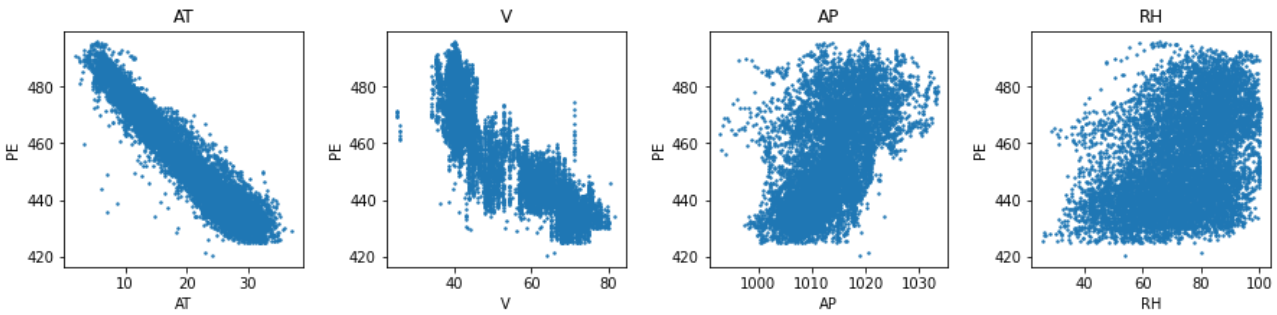

Figure 1 *Each input variable graphed against the output variable (PE).*

Performance Metric: Root Mean Squared Error and R^2

The two metrics used to evaluate the performance of each algorithm on this problem, Root Mean Squared Error (RMSE) and R^2. RMSE is calculated as:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - f(\vec{x}_i))^2},$$

where N is the sample size, y_i is the true value of the output variable, and $f(\vec{x}_i)$ is the model output as a function of the input vector (Lewis, p. 22). It can be interpreted as the square root of the mean squared difference between the actual value of the target variable and the model's prediction. Thus, it measures the

prediction error of the model in the actual units of the target variable, making interpretation of the model errors easier. Furthermore, it gives less weight to outliers than the Mean Squared Error (MSE). A perfect model that predicts the value of the target variable correctly for every feature vector would have a RMSE of zero. R^2 is defined as:

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i - f(\vec{x}_i))^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2},$$

where $\bar{y}$ is the mean of the target values y_i (Lewis, p. 23). R^2 is highly interpretable and indicates how much better the model performs relative to an unintelligent model that would simply predict the target variable mean for each instance. A value of 1 implies a perfect model.

Experimental Setup

The dataset constitutes 9,568 instances. 20% of the dataset was randomly selected to be set aside for testing of the final models and remains unseen throughout hyperparameter selection. The remaining 80% was used to train the model and select hyperparameters. To avoid selection bias associated with a single validation set, 10-fold cross validation was used to optimise model hyperparameters.

Two algorithms will be used to map the four input features to the target variable: a random forest regressor and a multi-layer perceptron regressor. Each models a non-linear relationship between model inputs and the output variable and is discussed in more detail below.

Multi-layer Perceptron Regressor

A multi-layer perceptron (MLP) is an example of a feedforward neural network that utilises an input layer, output layer and one or more hidden layers of neurons (Russell & Norvig, 2021, p. 729). Each neuron calculates a weighted sum of inputs from the previous layer and feeds this result into an activation function; the result of which is fed to each neuron in the next layer. A loss function, like MSE in the case of regression problems, is used to assess the model's performance and backpropagation is used to iteratively adjust the network weights. This is computationally expensive and so a mini-batch stochastic gradient descent algorithm is often used. To maximise computational efficiency and allow the MLP to model non-linear relationships, the Rectified Linear Unit (ReLU) function is selected to act as the activation function at each neuron.

In comparison to the simple univariate linear regression model described above, MLP is a discriminative model that can extract complex, non-linear relationships between features during training. However, unlike the baseline model, MLP has multiple hyperparameters that can be adjusted by the user to optimise its performance for a particular problem. Hyperparameters determine an algorithm's learning process and ultimately affect the weights that a model learns.

MLP has multiple hyperparameters, including the number of hidden layers, the number of neurons in each hidden layer, and the regularisation hyperparameter, alpha (α). There are several complex tuning methods that use probabilistic methods to select optimal values for hyperparameters. In this report, cross validation is used to adjust hyperparameters incrementally and values are selected to maximise the model's performance on unseen data. All three hyperparameters control model complexity and thus affect the model's ability to generalise to unseen data. Number of hidden layers and the number of neurons in each hidden layer determine the *number* of neurons, and α determines the *magnitude* of network weights (Hagan et al., 2014, p. 13-2). Due to the computational intensity of training the MLP for each permutation of the three hyperparameters noted above, the number of layers and neurons in each layer were optimised first, and the optimal value for α was determined using this setting. Additionally, the simplifying assumption that hidden layers constituted a uniform number of neurons was made to further reduce computational intensity.

There are four nodes in the input layer, one node in the output layer, and 6,888 training instances. To minimise model complexity, the maximum number of hidden layers selected was 3, and the maximum number of neurons in each hidden layer was set to 150. Performance of the model was assessed for increments of 1 hidden layer and 10 neurons, with a minimum complexity of 1 hidden layer of 10 neurons. Fig. 2 reveals that there is very little additional performance gained from increasing model complexity. The Ockham's razor

principle suggests that the higher the model complexity, the greater the possibility of errors (Hagan et al., 2014, p. 13-2). Thus, 1 hidden layer of 10 neurons was selected as the optimal network architecture for this problem.

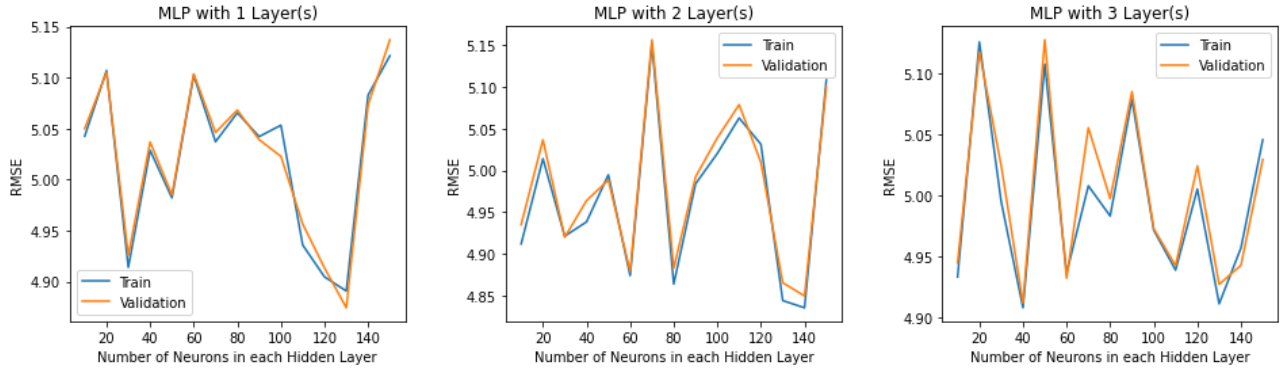

Figure 2 *Results of cross validation to select optimal number of hidden layers and number of neurons in each hidden layer.*

Cross validation for the regularisation hyperparameter, α, revealed unusual results. Larger α values reduce the size of model weights, thus reducing overfitting to the training set. However, as α increased, performance on the training set increased simultaneously. From Fig. 3, model performance on the validation set appears to be optimal for α = 40.

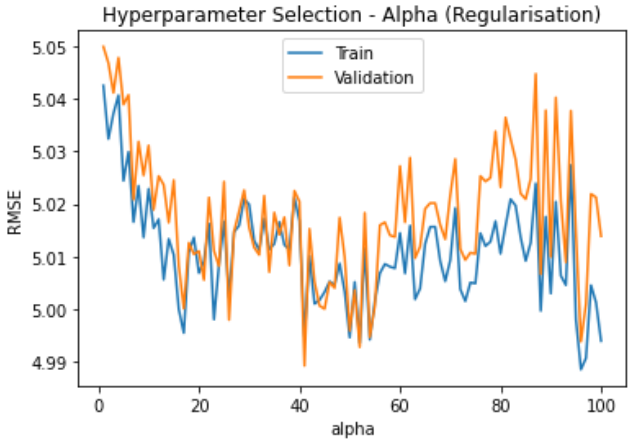

Figure 3 *Results of cross validation to select optimal value for alpha.*

Random Forest Regressor

Random forest models map input vectors to a target variable using multiple weak learner decision trees that make predictions based on multiple single feature decision boundaries. In line with the performance metric, RMSE, splits are determined to minimise mean squared error. An average value from all the decision tree outputs yields a single model prediction. The result is a complex, non-linear mapping from the feature vectors to the target variable. The hyperparameter, maximum tree depth (*D*), determines the maximum depth of each decision tree. Trees with greater depth tend to overfit the training set and better generalisation performance can be attained by specifying *D* (Lewis, p. 36). Other hyperparameters, including number of estimators (*T*) have little effect on model performance and generalisation, and so are set to sklearn's default values.

Cross validation is used to increment *D* from 1 to 20 and performance on the training and validation sets is examined to select an optimal value for the hyperparameter. Fig. 4 visualises the validation set performance plateauing at approximately *D* = 14. Thus, this was chosen as the optimal value for model prediction on the test set.

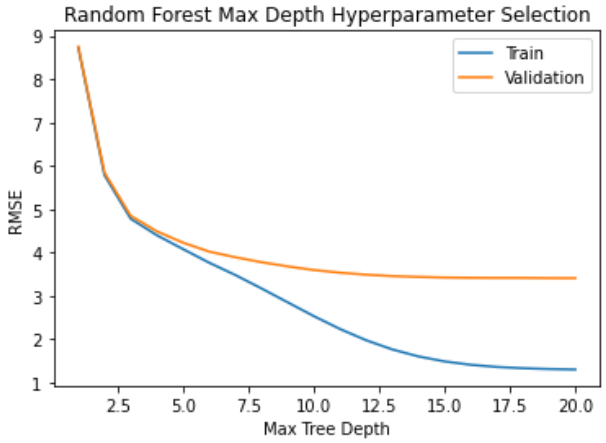

Figure 4 *Results of cross validation to select optimal maximum depth of each decision tree in the random forest.*

Results and Analysis

To compare the performance of each of these models on unseen data, the held-out test set, constituting 1,913 instances is used. The performance metrics, RMSE and R^2, are reported in the table below:

Model	**RMSE**	**R^2**
Univariate Linear Regression (Baseline)	5.374	0.902
Multi-layer Perceptron Regressor	4.961	0.917
Random Forest Regressor	3.288	0.963

Figure 5 *Results for each model's performance on the test set.*

The random forest regressor performs best on unseen data, achieving an RMSE of just 3.288 and R^2 of 0.963. Comparatively, the MLP only performs slightly better than the baseline model, achieving an RMSE that is only 7.7% smaller. The high R^2 achieved by the baseline univariate linear regression model implies that the problem can be competently modelled as a linear combination of the input variables (see Fig. 6). However, a more complex random forest model can extract more explanatory power using a non-linear mapping. Although an MLP model is suited to extracting complex relationships between features in a similar way, it is more suited to problems that involve large feature vectors. The random forest is much better suited to this problem, where there are a handful of informative features.

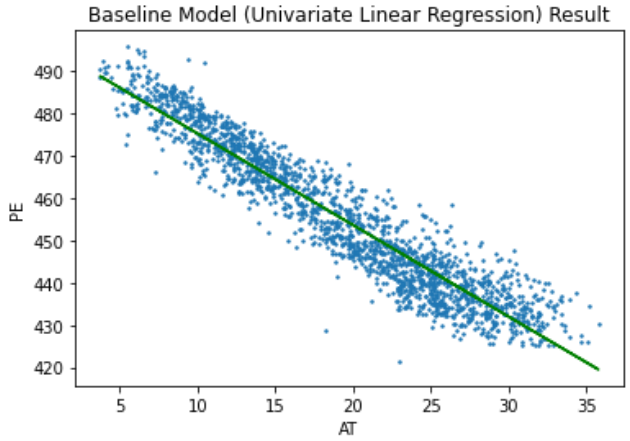

Figure 6 *Baseline model (linear regression) results on the test set.*

Due to their higher complexity, the random forest and MLP algorithms are more computationally expensive. Consequently, the decision to choose the random forest algorithm over the baseline for predicting PE would depend on stakeholders' preferences regarding performance and computational intensity/training time.

References

1. Hagan, M. T., Demuth, H. B., Beale, M. H., & De Jesús, O. (2014). Neural Network Design 2nd Edition. https://hagan.okstate.edu/NNDesign.pdf#page=469
2. Lewis, M. n.d. *Introduction to AI Week 13* [PowerPoint slides]. https://www.ole.bris.ac.uk/webapps/blackboard/content/listContent.jsp?course_id=_252253_1&content_id=_6968911_1&mode=reset
3. Lewis, M. n.d. *Introduction to AI Week 16* [PowerPoint slides]. https://www.ole.bris.ac.uk/webapps/blackboard/content/listContent.jsp?course_id=_252253_1&content_id=_6968916_1&mode=reset
4. Russell, S., & Norvig, P. (2021). Artificial Intelligence: a modern approach, ed. 3. Pearson Education Limited.