EMATM0044 Introduction to AI: Coursework Question 1

Student Number: 2315517

May 2023

1 Introduction

This report presents two solutions for the task of predicting the energy output of a power plant given four input features: Temperature (T), Ambient Pressure (AP), Relative Humidity (RH) and Exhaust Vacuum (V).

A labelled dataset was provided, which allowed supervised learning techniques to be employed. The output prediction consisted of a single *continuous* variable; Net hourly energy of the plant (PE), and therefor this problem was solved using regression models.

2 Methods

All code was implemented in Python using regression models provided by the library SciKit Learn [1]. Two regression models were chosen: K-Nearest Neighbours (KNN) and a Multi-Layer Perceptron (MLP) [2]. KNN selects the nearest K neighbours based on their feature distance and makes a prediction by combining the corresponding targets via a mean or weighted equivalent. MLP is a Neural Network approach, utilising a single hidden layer and trained via a back-propagation algorithm.

The provided data was split into two, with 80% allocated to training data and a separate 20% allocated to testing data. The models were trained entirely on the training data and final performance metrics calculated against the previously unseen testing data. Evaluating against previously unseen data ensures that the model performance is generalisable to new datasets.

2.1 Performance Metrics

The Mean Squared Error (MSE) [2] is a widely used approach for measuring the performance of regression models. It works by summing the squared distances from all predictions to their corresponding true targets. Squaring the distances ensures that positive and negative values are handled equally and gives a heavy weighting to outliers. These properties make MSE a suitable choice and thus it was chosen as the indicator of model prediction performance. MSE is calculated as follows:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

Where Y is the true target and $\hat{Y}$ is the prediction. This transforms the optimisation task into one of minimising MSE, where a perfect model would result in $MSE = 0$.

2.2 Hyperparameter Selection

Both regression models support a number of hyperparameters, which represent various configuration options. In order to determine robust hyperparameters, grid-searches across a range of possible values were performed. Overfitting was avoided through the utilisation of *K-Fold Cross-Validation* [3], splitting the training data into 20 folds, where each fold corresponds to a training set and independent validation set. Each model was trained for a particular hyperparameter configuration using the training set, and associated prediction performance metrics were gathered against the previously unseen validation set. The mean average MSE was then calculated over the 20 validation sets as a representative score for that configuration of hyperparameters. This approach ensured the hyperparameters selected for the model are demonstrably robust and that the associated model performance is generalisable to previously unseen data.

2.2.1 K-Nearest Neighbours

Two hyperparameters were evaluated for the KNN regression model. The first to consider is K; the number of neighbours. This directly controls the number of neighbouring data points in the training set that are used to predict a new target. The range of valid values is $1 \leq K \leq training\_set\_size$. At the extreme lower end, a k value of 1 would cause the prediction to be identical to the single closest neighbour. This would likely give very noisy (and therefor uncertain) predictions. At the other extreme, a value equal to the size of the training set would calculate the overall mean, and be equivalent to the aforementioned baseline. An optimal value lies somewhere between the two. The second hyperparameter considered was *weights*; which specifies how much contribution each of the neighbours makes to the prediction. $uniform$ indicates that all neighbours make the same contribution to the predicted target, whereas $distance$ indicates that the contribution should be inversely proportional to the distance.

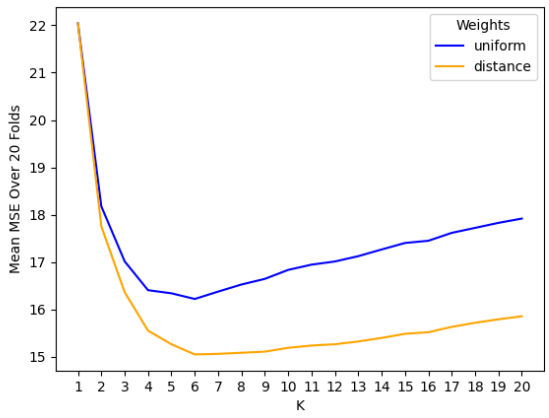

Figure 1: KNN Hyperparameter Performance

A two dimensional grid search was performed over thse two hyperparameters, where $1 \leq K \leq 20$ and $weights \in \{uniform, distance\}$. The corresponding MSE is presented in Figure 1. The lowest MSE was associated with $K = 6$ and $weights = distance$, which were thus selected for use in the final model. Although the grid search was only applied to a relatively small range of possible values of K, there is a high chance it is the optimal value, since averaging over large numbers of dispersed neighbours will likely reduce individual prediction accuracy further.

2.2.2 Multi-Layer Perceptron

Two hyperparameters were also evaluated for the MLP regression model. The first, the *activation function*, is often required in order to introduce some non-linearity to the model. A value of *identity* indicates that no function is applied. Thus for linear problems *identity* is sufficient, and for non-linear problems *tan* or *logisitic* would be expected to perform better. Another hyperparameter is the number of nodes in the hidden layer, which can be tuned in order to increase or decrease the model complexity. Too simple a model will be unable to adequately capture the relationship between inputs and outputs, whereas too complex a model can be more difficult to train.

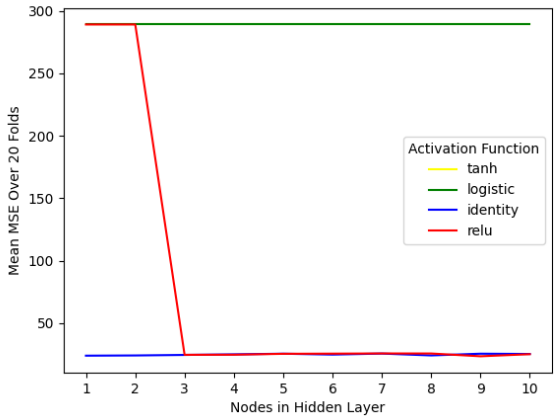

Figure 2: MLP Hyperparameter Performance

A two-dimensional grid search was performed over these two hyperparameters, where $activation\_function \in \{tanh, logistic, identity, relu\}$ and $1 \leq number\_of\_nodes \leq 10$. The corresponding MSE can be found in Figure 2 (note that $tanh$ is hidden behind $logistic$ and can be assumed equivalent). The $tanh$ and $logistic$ functions are calculated via the trigonometric functions of the same names, and $relu$ via the rectified linear unit function, where $f(x) = max(0, x)$. The strong performance of *identity* indicates that the problem is highly linear, and this is further confirmed by the lack of significant affect from varying the number of nodes. Since they were associated with the lowest MSE, $activation\_function = identity$ and $number\_of\_nodes = 1$ were selected for use in the final model. These values effectively reduce the MLP to something like a linear regression model. Due to the strong indications that this problem is linear, a simple

Linear Regressor was also added to the results section for comparison.

Also of note was the use of the *lbfgs* solver rather than *adam* or *sgd*, since that was expected to converge faster and to more optimal solutions when using small data sets [4].

2.3 Baseline

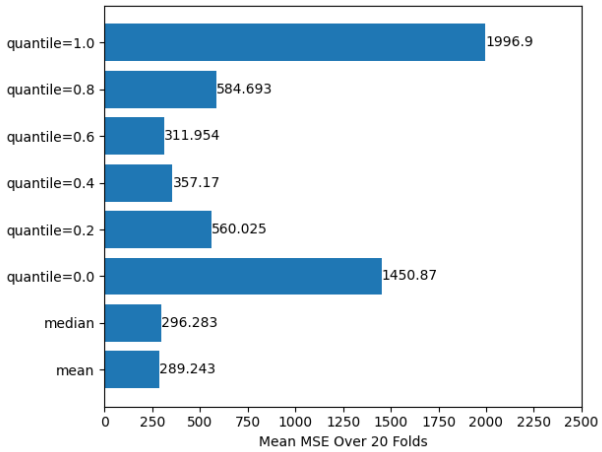

Figure 3: Baseline Performance

For comparison purposes a baseline model was employed which always predicted the mean average of the training data target. This was deemed a suitable choice as MSE heavily punishes outliers, and thus a model which always predicts the mean should give a better score than one which predicted any other single value (e.g. extremes such as min or max). In order to confirm this idea a grid search was performed over a range of prediction strategies, where $strategy \in \{mean, median, quantile\}$ and $0.0 \leq quantile \leq 1.0$ in increments of 0.2. Using the same K-Fold Cross Validation techniques presented in the hyperparameter selection, the corresponding MSE were generated and shown in Figure 3. Since *mean* gives the lowest MSE, as expected, it was selected as an appropriate baseline.

3 Results & Analysis

The following results were produced by training the models on the entire training data, with the previously presented hyperparameters, and making predictions against the entire previously unseen testing data.

Firstly, a visual inspection of predictions by feature can be performed in order to gain some understanding of both the problem and model performance. It can be seen in Figures 4 and 5 that the problem is indeed highly linear and that the output target PE is particularly strongly inversely correlated with feature T. Furthermore, it can be seen that both the KNN and MLP models achieved a high level of success in predicting PE, as it is not easy to visually distinguish their predictions from the true data.

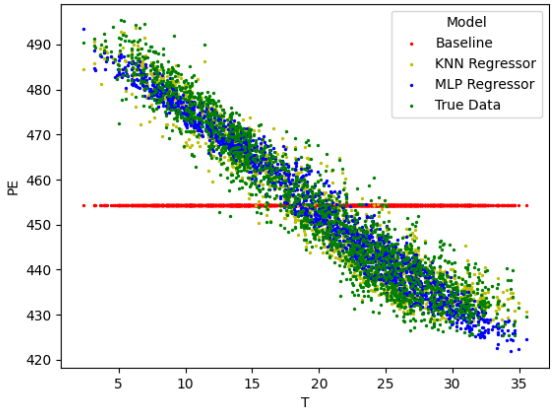

Figure 4: Feature T vs. PE

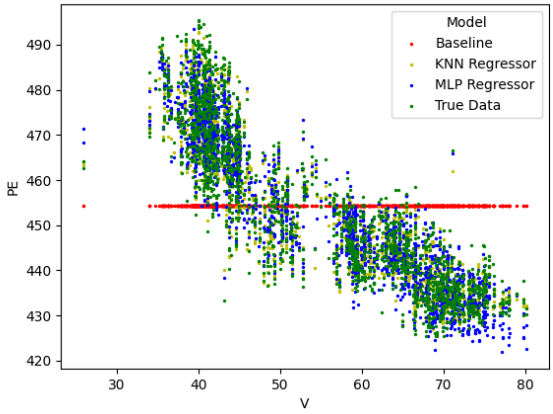

Figure 5: Feature V vs. PE

Feature	T	V	AP	RH
Weight	1.79	0.22	-0.06	0.14

Table 1: Trained MLP Connection Weights

The MLP model provides access to the weights of the trained model, and these are shown in Table 1. This illustrates that the model has learnt to heavily bias it's prediction towards feature T, with by far the largest weighting, again confirming the previous observation that PE is most strongly correlated with T.

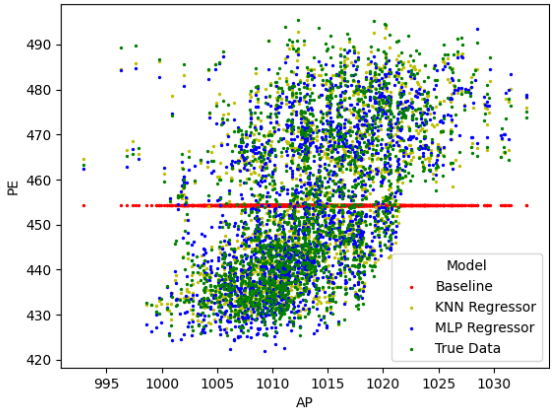

Figure 6: Feature AP vs. PE

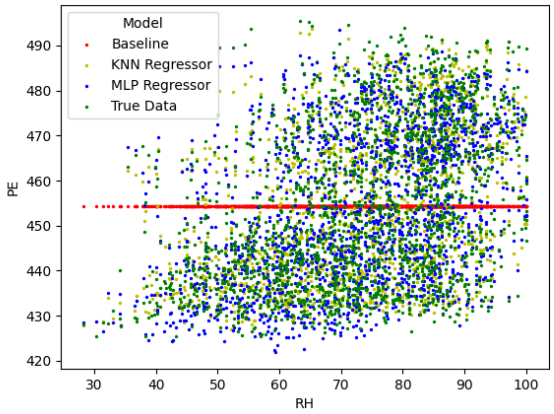

Figure 7: Feature RH vs. PE

The data in Figures 6 and 7 is less clear, with a much wider distribution and weaker correlation between the features and target PE. However, again it is not easy to distinguish the model predictions from the true data, which indicates the models achieved a good level of prediction performance.

The final comparison of model performance can be found in Figure 8. All of the models performed significantly better than the baseline, with the KNN model performing best overall. This quantitatively confirms the observation from the previous graphs, where the regression models were able to predict targets approximately equivalent to the true data. As expected the MLP performed equivalently to a simple Linear model due to its hyperparameter configuration. The superior performance of the KNN over the MLP and Linear models indicates that there might be some non-linearity in the problem, which the KNN was successfully able to capture.

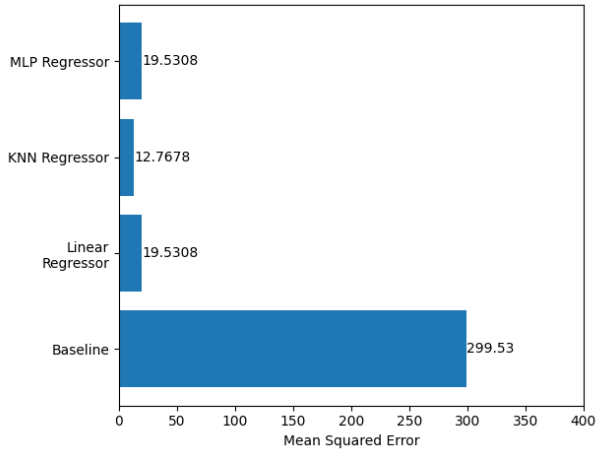

Figure 8: Performance Comparison by MSE

In summary the problem was shown to be approximately linear, with T being the most significant feature and the KNN model giving the highest accuracy predictions.

References

[1] A. G. Fabian Pedregosa, Gael Varoquaux and V. Michel, "Scikit learn," https://scikit-learn.org/stable/, 2023, [Online; accessed May 2023].

[2] S. J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 4th ed. Pearson Education, 2021.

[3] T.-T. Wong and P.-Y. Yeh, "Reliable accuracy estimates from k-fold cross validation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, no. 8, pp. 1586–1594, 2020.

[4] Q. V. Le, J. Ngiam, A. Coates, A. Lahiri, B. Prochnow, and A. Y. Ng, "On optimization methods for deep learning," in *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ser. ICML'11. Madison, WI, USA: Omnipress, 2011, p. 265–272.