Effective Tuning of Regression Models using an Evolutionary Approach: A Case Study

Seamus Lankford
Adapt Centre, Dublin City University, Ireland
seamus.lankford@adaptcentre.ie

ABSTRACT

Hyperparameters enable machine learning algorithms to be customized for specific datasets. Choosing the right hyperparameters is a challenge often faced by machine learning practitioners. With this research, tuning of hyperparameters for regression models was explored. Models predicting house prices in King County were created using a detailed suite of regression algorithms. Traditional approaches, and evolutionary algorithms, for improving model accuracy were evaluated. A variety of feature selection methods and hyperparameter tuning using grid search, random search and pipeline optimization were also studied as part of the traditional approaches. Furthermore, evolutionary algorithms were applied to model optimization. In this paper, it is shown that an evolutionary approach, implemented with TPOT, achieves the highest accuracy for a regression model based on the King County dataset. Regarding metrics, combining the RMSE and R^2 metrics is shown to be an effective means of determining model accuracy. Finally, greedy feature selection performed best when a variety of feature selection methods are compared.

CCS CONCEPTS

• **Computing Methodologies**; • **Machine Learning**; • **Machine Learning Algorithms**;

KEYWORDS

Grid Search, Random Search, Feature Selection, Auto ML, Genetic Algorithm, TPOT

ACM Reference Format:
Seamus Lankford. 2020. Effective Tuning of Regression Models using an Evolutionary Approach: A Case Study. In *2020 3rd Artificial Intelligence and Cloud Computing Conference (AICCC 2020), December 18–20, 2020, Kyoto, Japan.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3442536.3442552

1 INTRODUCTION

To evaluate the effective tuning of regression model parameters, models were developed using a dataset provided by King County, Washington, USA [1]. The dataset contains detailed records of homes which were sold in the King County area of Washington between May 2014 and May 2015. There are 21,613 observations in the dataset which contains 20 house features plus the target regression value price.

The objective of the research was to investigate all aspects of building a predictive model and to use these findings in creating the optimal regression model for predicting house prices in the King County area. Therefore, several key areas were explored namely pre-processing techniques, scaling methods, choice of regression models, feature selection techniques and hyperparameter tuning. In addition, hyperparameter tuning using pipelines and genetic algorithms was explored.

2 BACKGROUND

Automated Machine Learning (AutoML) [2] generates optimized models automatically when provided with datasets, thus reducing the need for data scientists. AutoML tools, including those that use genetic programming, were investigated as part of this research. Well known AutoML tools include Auto-WEKA [3], Hyperopt-Sklearn [4], AutoKeras [5], Auto-Sklearn [6, 7] and TPOT [8].

2.1 Hyperparameter Optimization

In order to customize machine learning models to particular datasets, hyperparameters are employed. The performance of standard machine learning libraries can be improved through hyperparameter optimization (HPO) of its default settings [9, 10]. Furthermore, HPO reduces the human effort required for implementing machine learning [11].

Grid search is a tuning technique which calculates the best hyperparameter values through an exhaustive search. The search is performed on a user defined set of parameter values [12]. However, such an approach suffers from the curse of dimensionality in that as the number of features grows, the amount of data needed to generalize accurately grows exponentially [13]. An effective and less computationally intensive alternative to grid search is to use random search [14] which samples random configurations. In the context of this research, both grid search and random search are evaluated.

2.2 TPOT

TPOT is a python based automated machine learning tool that optimizes pipelines using genetic programming. Aspects of machine learning are automated by exploring different pipelines to identify the best fit for the model's data. TPOT considers multiple machine learning algorithms such as Random Forest (RF), linear models and SVM in a pipeline with different pre-processing steps including missing value imputation, scaling, PCA and feature selection. In addition, the model hyperparameters are selected.

AICCC 2020, December 18–20, 2020, Kyoto, Japan

ACM ISBN 978-1-4503-8883-2/20/12…$15.00
https://doi.org/10.1145/3442536.3442552

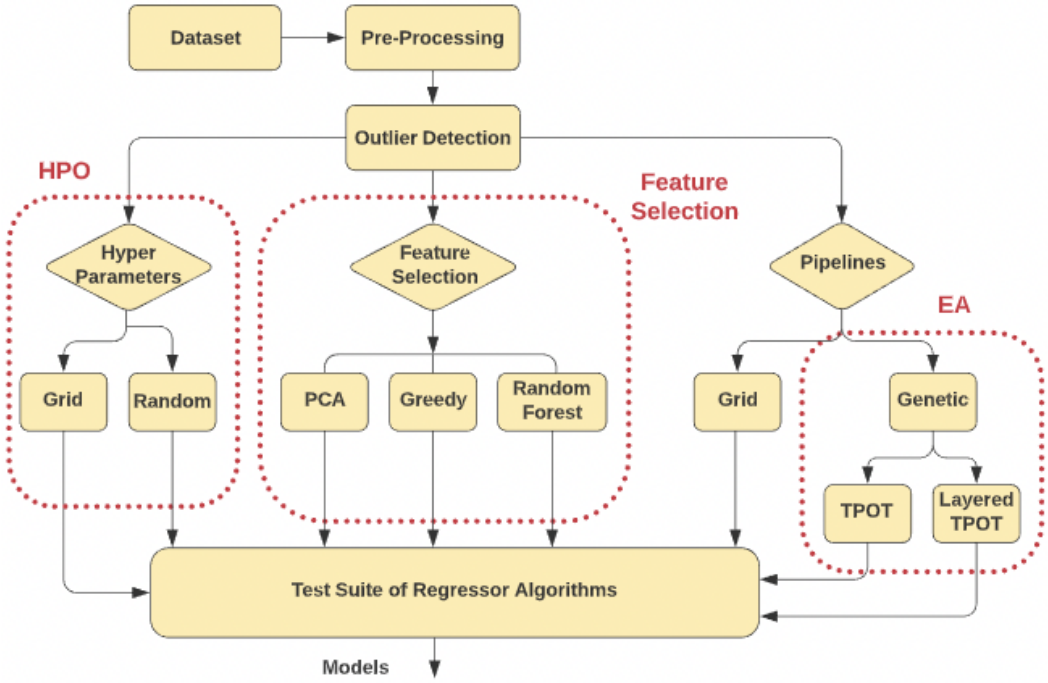

Figure 1: System Overview

Layered TPOT is a modification to the TPOT approach which accelerates pipeline optimization. This approach is explored by Gijsbers et al [15]. The adopted approach aspires to producing pipelines equally as good as the original in significantly less time. In the modified approach to TPOT, time is reduced by initially evaluating the pipelines on a small subset of the data, and only allowing promising pipelines to be evaluated on the full dataset. As an algorithm based on genetic programming, TPOT was chosen for its ease of use coupled with the fact that it is widely used in the research community.

2.3 Tuning Evolutionary Parameters

Finding appropriate parameter values for evolutionary algorithms (EA) is one of the challenges experienced in the field of evolutionary computing. The effect of parameter tuning on EAs is discussed in some detail by Eiben and Smit [16] in their general framework of an evolutionary algorithm. The results show that population size is the most significant control parameter for improving model accuracy. This finding is consistent with the experiences of other researchers [17] in their work on efficient genetic algorithms for improving optimization.

In developing a predictive model for the King County dataset, the application both implements TPOT and adheres to the key elements of the general framework of Eiben and Smit. While the implementation does not allow control of all parameters, it does enable the key parameters to be set.

3 APPROACH

Multiple pre-processing techniques were applied when evaluating the performance of a range of models. Furthermore, a comprehensive suite of feature selection methods was investigated. Selecting different functions for scaling has significant effects on model accuracy. It is difficult to anticipate, in advance, which scaling method is most suitable. Therefore, different scaling methods were applied depending on user specified parameter values. The scaling methods of Min Max Scaler, Standard Scaler and Normalization are made available and set as parameters specified when invoking the application.

Modelling of datasets, using machine learning, provides a wide array of choices. It is difficult to anticipate which of these algorithms will perform best for the model under development without running adequate tests. A regressors array was used as a parameter for all functions which tested regression models.

4 DESIGN

Key components of the system are highlighted in Figure 1. Hyperparameter optimization, with both grid search and random search, was implemented and the optimum parameters were selected. Building on this, suitable pipelines were recommended. Regression algorithms were tested with and without feature selection. The feature selection techniques employed were PCA, Random Forest (RF) and greedy feature selection.

TPOT, and the faster approach of Layered TPOT were used to implement the genetic algorithm which searched for optimal machine learning pipelines.

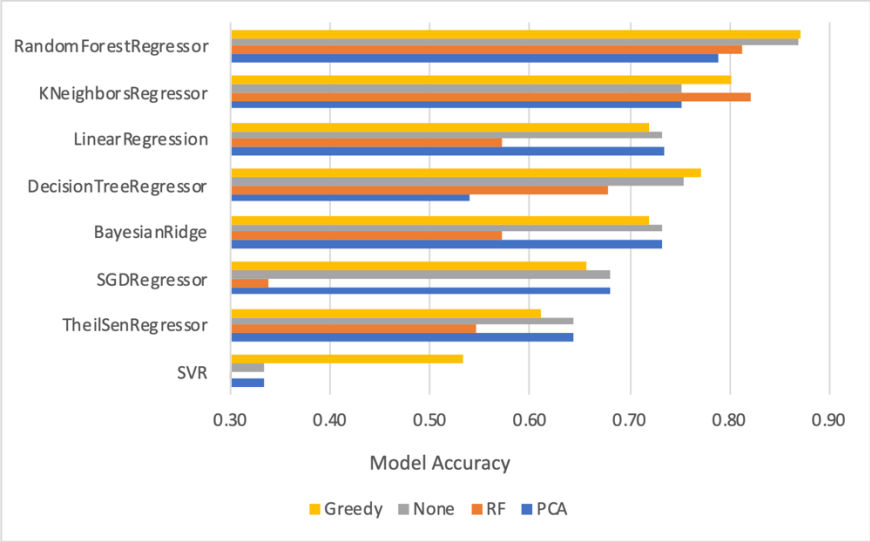

Figure 2: Feature Selection Impact on Accuracy

Table 1: Regression Model Performance without Feature Selection

Algorithm	R^2	RMSE
Random Forest Regressor	0.869	0.016
Decision Tree Regressor	0.753	0.022
KNeighbors Regressor	0.752	0.020

5 EVALUATION

5.1 Metrics

Several metrics were evaluated in determining model performance namely : R^2 coefficient, Mean Absolute Error (MAE), Mean Squared Error (MSE) and Root Mean Squared Error (RMSE).

In assessing the performance of various models, RMSE was selected, instead of MAE and MSE since it gives a better measure of fitness. RMSE gives a high weighting to large errors since errors are squared before they are averaged:

$$RMSE = \sqrt{{}^{1}/_{n} \sum_{i=1}^{n} (S_i - O_i)^2}$$

where S_i are the predicted values and O_i are the observations and n is the number of observations.

The R^2 coefficient compares the performance of a model on a test set with the performance of an imaginary model that always predicts the average values from the test set.

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (y - \hat{y}_i)^2}{\sum_{i=1}^{n} (y_i - \bar{y}_i)^2}$$

where y is the actual value, $\hat{y}_i$ is the predicted value, $\bar{y}_i$ is the mean of the y values and n is the number of observations.

5.2 Regression Model Performance

A combination of metrics is often required to assess model performance [18]. A combination of R^2 with RMSE was used to determine algorithm performance. Algorithms which demonstrate a high R^2 and a low RMSE should perform well. The performance of a large suite of regression models was tested. A summary of the results, using the R^2 and RMSE metrics, is shown in Figure 2 and in Figure 3

The best performing algorithms are extrapolated from these results and presented in Table 1. The RF Regressor is the best performing algorithm across both key metrics of R^2 and RMSE.

5.3 Feature Selection

A broad range of feature selection techniques, and their impact on model performance, was evaluated. The feature selection techniques examined in detail included PCA selection, tree based selection and greedy backward selection.

5.3.1 PCA Feature Selection. It can be seen, from Figure 2 and Figure 3, that PCA adversely affects R^2 and RMSE scores across the best performing regressor models. Therefore, as a method of reducing dimensionality, it works well but this comes at a high cost of significantly reducing model accuracy across the key metrics. For example, the R^2 coefficient for the RF Regressor is 0.869 without PCA and this drops to an R^2 of 0.79 when PCA is employed. Given this cost, PCA is not recommended for this dataset. These results are supported by the grid search experiments, the pipeline evaluations and the genetic algorithms, none of which recommended using PCA.

5.3.2 Random Forest Feature Selection. The experimental findings, summarized in Figure 2 and in Figure 3, compare model performance on all algorithms using the R^2 and RMSE metrics.

The best performing model was KNeighbors Regressor which had an R^2 of 0.821 and RMSE of 0.019, after RF feature selection,

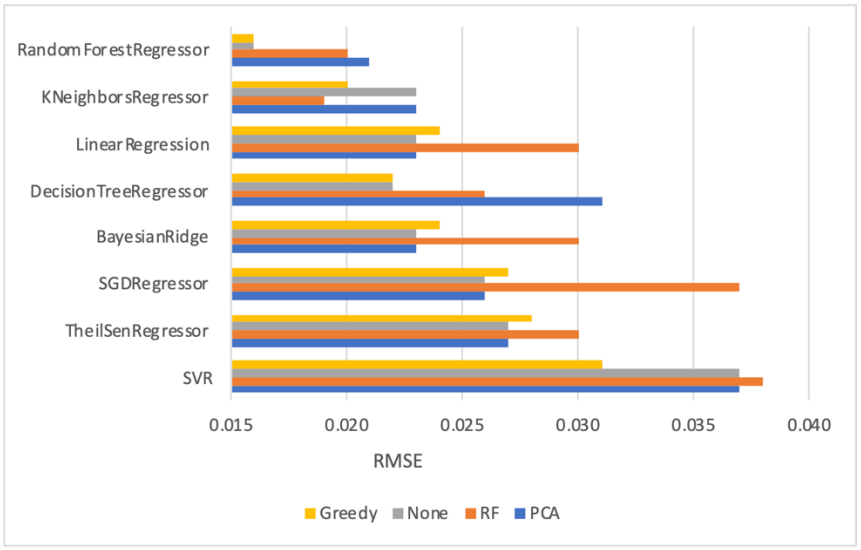

Figure 3: Feature Selection Impact on RMSE

compared to R^2 of 0.751 and RMSE of 0.023, prior to RF feature selection. Clearly RF feature selection led to an improvement in model accuracy when KNeighors Regressor is used as the model's algorithm.

However, from Table 1, it can be seen that the R^2 and RMSE results of the top performing model without feature selection was 0.869 and 0.016, respectively, for the RF Regressor. These metrics are significantly better than the KNeighbors Regressor with an R^2 of 0.821 and RMSE of 0.019 with RF feature selection. Therefore, feature selection using random forest feature selection is not advised for the King County dataset.

5.3.3 Greedy Feature Selection. The results presented in Figure 2 and in Figure 3 are based on greedy selection using a Passive Aggressive Regressor estimator and the KNeighbors Regressor model.

In comparing greedy feature selection with both PCA and random forest feature selection, it can be seen that the greedy algorithm performs best. Top model accuracy, where greedy selection was initially applied on the dataset, was better than in cases where greedy selection had not been applied. The associated R^2 score of the best performing model, with greedy selection, is higher than the R^2 score achieved on running the regressors without feature selection.

There are two components in the greedy algorithm which determine how effective greedy feature selection is: the estimator and the model. An array of regressor algorithms was used to create an exhaustive set of (estimator, model) combinations. Each combination applied greedy feature selection to the King County dataset.

With this approach, the greedy algorithm was run on estimator / model combinations to see their effect on model performance.

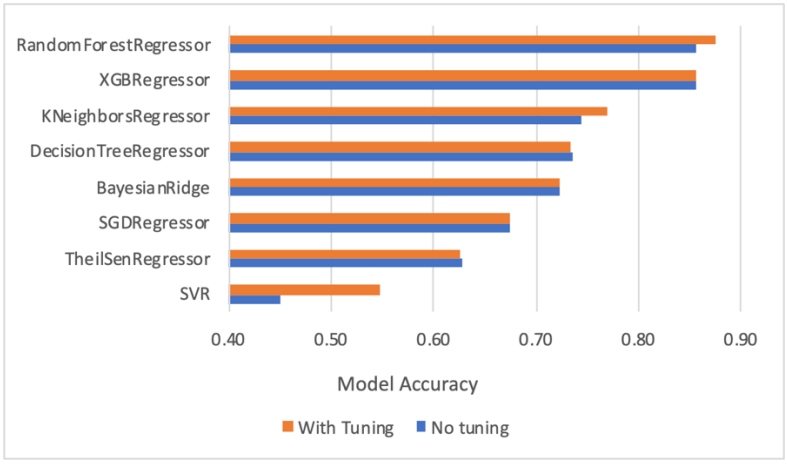

Figure 4: Grid Search Impact on Accuracy

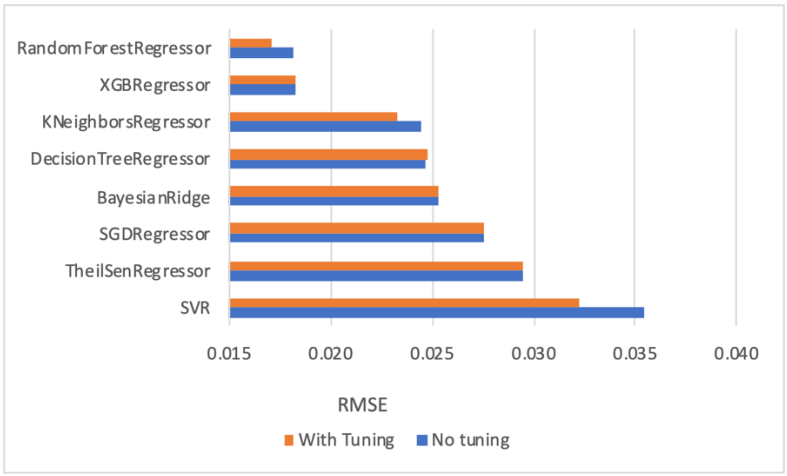

Figure 5: Grid Search Impact on RMSE

Using a Passive Aggressive Regressor as estimator and a RF Regressor as the model, feature selection was initially performed on the dataset. The transformed data was then used by all algorithms in the regressors array. The RF Regressor was the best performing regressor with an R^2 of 0.88.

Therefore, it was shown that feature selection can improve model performance and it is recommended to choose a greedy strategy for feature selection on the King County dataset.

5.4 Hyperparameter Optimization

The approach taken to the evaluation of hyperparameters is consistent with how previous evaluations are conducted. An array of regressor algorithms is used and the parameters specific to that algorithm are applied. The appropriate models are subsequently built and assessed. The full test run contains the R^2, MAE, MSE and RMSE as evaluation metrics. In the graphical depiction of the test results, model accuracy is displayed using the key metrics of R^2 and RMSE.

5.4.1 Optimization with Grid Search. Figures 4 and 5, illustrate that grid search improved or maintained model accuracy, for all algorithms. The best performing model, with its tuned parameters is the Random Forest Regressor using n estimators of 100 and a max depth of 20 achieving an R^2 of 0.8724.

5.4.2 Optimization with Random Search. The results, from Figures 6 and 7, illustrate that random search improved model accuracy, or at a minimum maintained accuracy, for all algorithms. The best performing model using random search, with its tuned parameters is the Random Forest Regressor with n estimators of 91 and a max depth of 22 achieving an R^2 of 0.8717. On comparing the performance of grid search with random search it can be seen there is little difference in terms of model accuracy with both approaches. Such a finding is consistent with the literature [14]. Grid search is marginally better at hyperparameter tuning for model accuracy. The best parameters found for grid search yielded an R^2 of 0.8724 whereas random search has an R^2 of 0.8717.

Table 2: Pipeline Accuracy using TPOT (full dataset)

Test	Pop	Gen	Pipelines	R^2	Time / s
1	10	5	50	0.8969	367
2	20	10	200	0.8914	1846
3	200	10	2000	0.9090	29394

5.4.3 Optimization with Pipelines. Hyperparameter optimization through pipelines was implemented as part of the research. Pipelines were constructed using SVM, a DT regressor and a RF regressor. The model with the highest R^2 coefficient used the RF Regressor algorithm. While hyperparameter optimization, using pipelines, produced good R^2 values (RF: 0.83, DT: 0.737), it is worth noting that it is considerably lower than the R^2 (0.869) achieved using a Random Forest without feature selection, as illustrated in Table 1

5.5 Optimization with Genetic Pipelines

The performance of TPOT was evaluated against traditional approaches to machine learning. To accelerate model development, a layered TPOT approach using 25% of the King County dataset, was adopted. The results, displayed in Table 2 and Table 3, support the findings of Gijsbers [15]. While the R^2 accuracy achieved on the reduced dataset is lower than the full dataset, the difference is trivial for higher population sizes. Using default crossover and mutation parameters with a population size of 200 and 10 generations, 2000 pipelines were evaluated for both the full and reduced datasets. The best performing pipeline within the full dataset built a model with an R^2 accuracy of 0.9090 in a time of 489 minutes. By comparison, the best performing pipeline using the reduced dataset achieved model accuracy of 0.8859 in a time of just 44 minutes illustrating that simple TPOT configurations work well. In fact, it has been shown that a simple TPOT configuration provides greater accuracy than an exhaustive grid search or random search. A TPOT configuration with a population size of 10, 5 generations, mutation

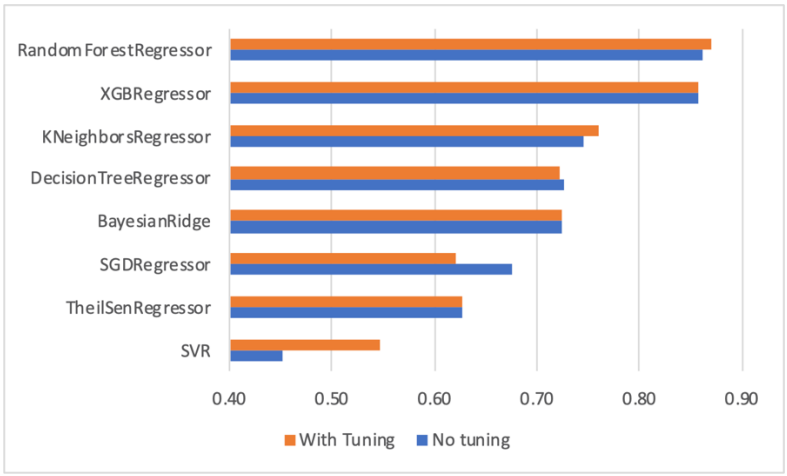

Figure 6: Random Search Impact on Accuracy

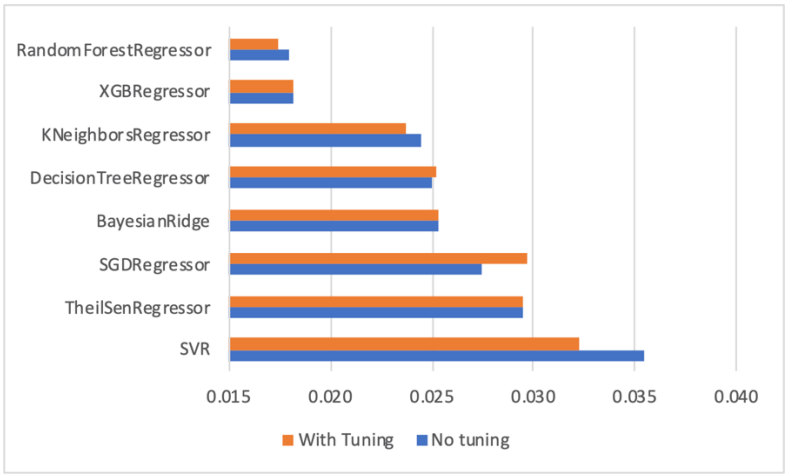

Figure 7: Random Search Impact on RMSE

Table 3: Pipeline Accuracy using layered TPOT (25% of original dataset)

				Mut 0.9 Cross 0.1		Mut 0.45 Cross 0.1		Mut 0.3 Cross 0.1	
Test	**Pop**	**Gen**	**Pipelines**	**R^2**	**Time / s**	**R^2**	**Time / s**	**R^2**	**Time / s**
1	10	5	50	0.8476	86	0.8476	115	0.8390	430
2	20	10	200	0.8396	500	0.8330	584	0.8390	326
3	200	10	2000	0.8859	2683	0.8471	989	0.8524	623

(0.9) and crossover (0.1) achieved a model accuracy of 0.8969 in 6 minutes compared with grid search, which achieved an R^2 model accuracy of 0.874 in 34 minutes.

6 CONCLUSION

The Random Forest Regressor was identified as the top performing algorithm on initial test runs carried out without feature selection, grid optimization, pipeline optimization or the use of genetic algorithms.

Feature data, which is PCA transformed, is unsuitable for improving the model accuracy of the King County dataset. Random

forest selection was an improvement over PCA but ultimately backward greedy selection both reduced dimensionality and improved model accuracy. The best performing estimator/model combination for greedy feature selection, in the case of this dataset, is the Passive Aggressive Regressor as estimator using a Random Forest Regressor as the model. The transformed dataset using this optimal combination achieved an R^2 accuracy of 0.88 which illustrated that feature selection is a useful tool in improving the accuracy.

Hyperparameter tuning using grid search and random search was investigated. Even though the improvements were only marginal, both approaches improved model accuracy. Running a Random Forest Regressor, without hyperparameter tuning yielded an R^2 of 0.869. In fact, it is interesting to note that running a Random Forest Regressor on a dataset, transformed using greedy feature selection, provided better accuracy results (R^2 of 0.88) than grid search.

The use of genetic algorithms ultimately led to the greatest improvements in model accuracy. A genetic search algorithm, as a TPOT implementation, identified the most suitable pipeline enabling an R^2 of 0.909 using the full dataset. This represents a 4% increase in accuracy when compared with the non-evolutionary approach of a Random Forest Regressor, which achieved a score of 0.869. More complex genetic configurations, with higher population sizes and a greater number of generations does not always lead to better solutions due to the stochastic nature of such algorithms. Using the full dataset, test run 2 generated a marginally lower R^2 coefficient (0.8914) compared with test run 1 (0.8969) even though it was four times more complex and took 5 times longer to complete. Genetic algorithms are good at identifying algorithms that would not have previously been considered. In the cases of test run 1 and test run 2, with the full dataset, the XGB Regressor algorithm was identified as optimal. This algorithm was not initially considered when modelling the King County dataset.

REFERENCES

[1] Harlfoxem. House Sales in King County, USA. Retrieved 25 Aug, 2016 from https://www.kaggle.com/harlfoxem/housesalesprediction

[2] Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren. 2019. Automated machine learning: methods, systems, challenges. Springer.

[3] Lars Kotthoff, Chris Thornton, Holger H Hoos, Frank Hutter, and Kevin Leyton-Brown. 2017. Auto-WEKA 2.0: Automatic model selection and hyperparameter optimization in WEKA. The Journal of Machine Learning Research 18, 1 (2017), 826–830.

[4] Brent Komer, James Bergstra, and Chris Eliasmith. 2019. Hyperopt-Sklearn. In Automated Machine Learning. Springer, Cham, 97–111.

[5] Haifeng Jin, Qingquan Song, and Xia Hu. 2019. Auto-keras: An efficient neural architecture search system. In Proceedings of the 25 th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 1946–1956.

[6] Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Springenberg, Manuel Blum, and Frank Hutter. 2015. Efficient and robust automated machine learning. In Advances in neural information processing systems. 2962–2970.

[7] Matthias Feurer, Katharina Eggensperger, Stefan Falkner, Marius Lindauer, and Frank Hutter. 2020. Auto-sklearn 2.0: The next generation. arXiv preprint arXiv:2007.04074 (2020).

[8] Randal S Olson and Jason H Moore. 2019. TPOT: A Tree-Based Pipeline Optimization Tool for Automating. Automated Machine Learning: Methods, Systems, Challenges (2019), 151.

[9] Randal S Olson, William La Cava, Zairah Mustahsan, Akshay Varik, and Jason H Moore. 2017. Data-driven advice for applying machine learning to bioinformatics problems. arXiv preprint arXiv:1708.05070 (2017).

[10] Samantha Sanders and Christophe Giraud-Carrier. 2017. Informing the use of hyperparameter optimization through metalearning. In 2017 IEEE International Conference on Data Mining (ICDM). IEEE, 1051–1056.

[11] Matthias Feurer and Frank Hutter. 2019. Hyperparameter optimization. In Automated Machine Learning. Springer, Cham,3–33.

[12] Douglas C Montgomery. 2017. Design and analysis of experiments. John Wiley & Sons.

[13] Ezekiel T Ogidan, Kamil Dimililer, and Yoney Kirsal Ever. 2018. Machine learning for expert systems in data analysis. In 2018 2nd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT). IEEE, 1–5.

[14] James Bergstra and Yoshua Bengio. 2012. Random search for hyper-parameter optimization. The Journal of Machine Learning Research 13, 1 (2012), 281–305.

[15] Pieter Gijsbers, Joaquin Van-schoren, and Randal S Olson. 2018. Layered TPOT: Speeding up tree-based pipeline optimization. arXiv preprint arXiv:1801.06007 (2018).

[16] Agoston Endre Eiben and Selmar K Smit. 2011. Evolutionary algorithm parameters and methods to tune them. In Autonomous search. Springer, 15–36.

[17] Ali Alajmi and Jonathan Wright. 2014. Selecting the most efficient genetic algorithm sets in solving unconstrained building optimization problem. International Journal of Sustainable Built Environment 3,1 (2014), 18–26.

[18] Tianfeng Chai and Roland R Draxler. 2014. Root mean square error (RMSE) or mean absolute error (MAE)? – Arguments against avoiding RMSE in the literature. Geoscientific model development 7, 3 (2014), 1247–1250